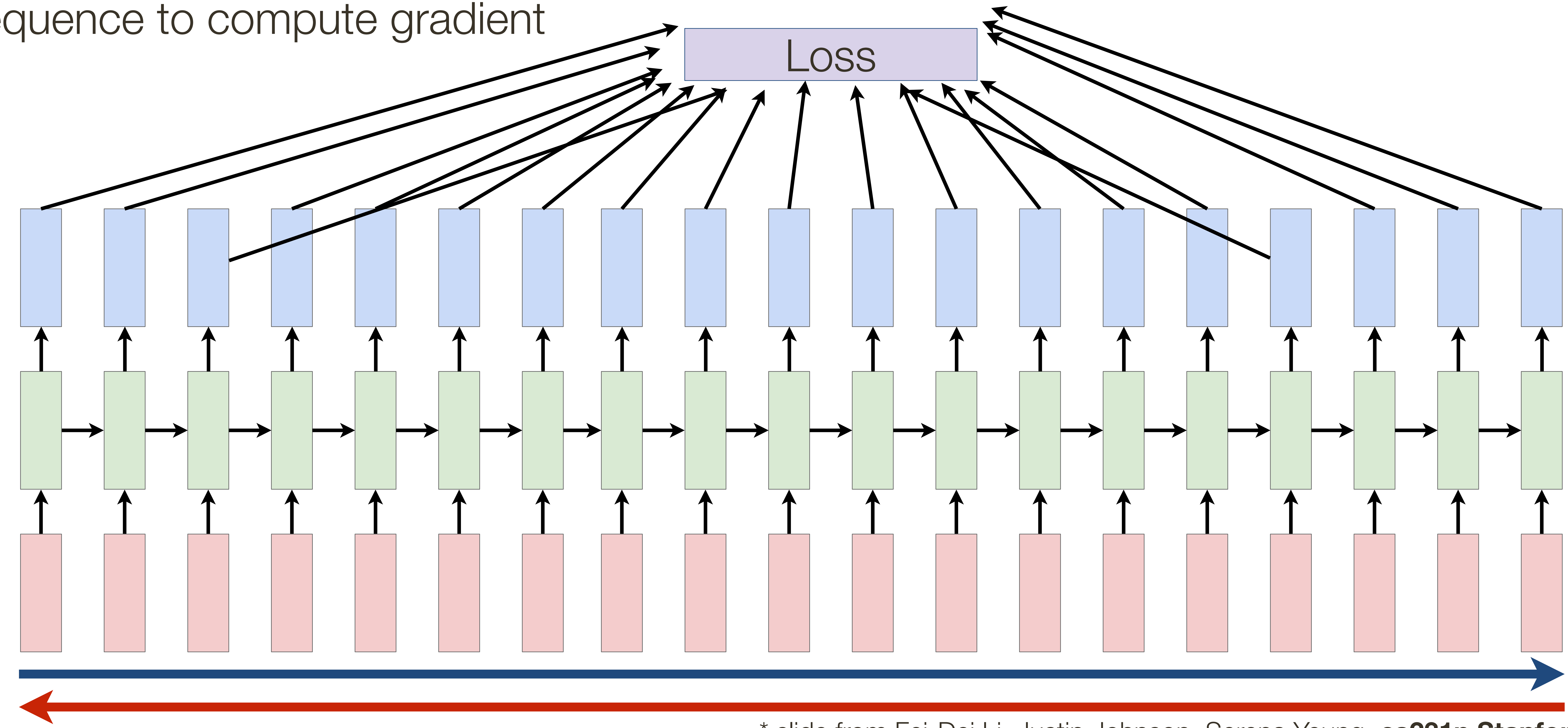
THE UNIVERSITY OF BRITISH COLUMBIA

Topics in AI (CPSC 532S):
Multimodal Learning with Vision, Language and Sound

Lecture 9: RNNs (part 2) + Applications

BackProp Through Time

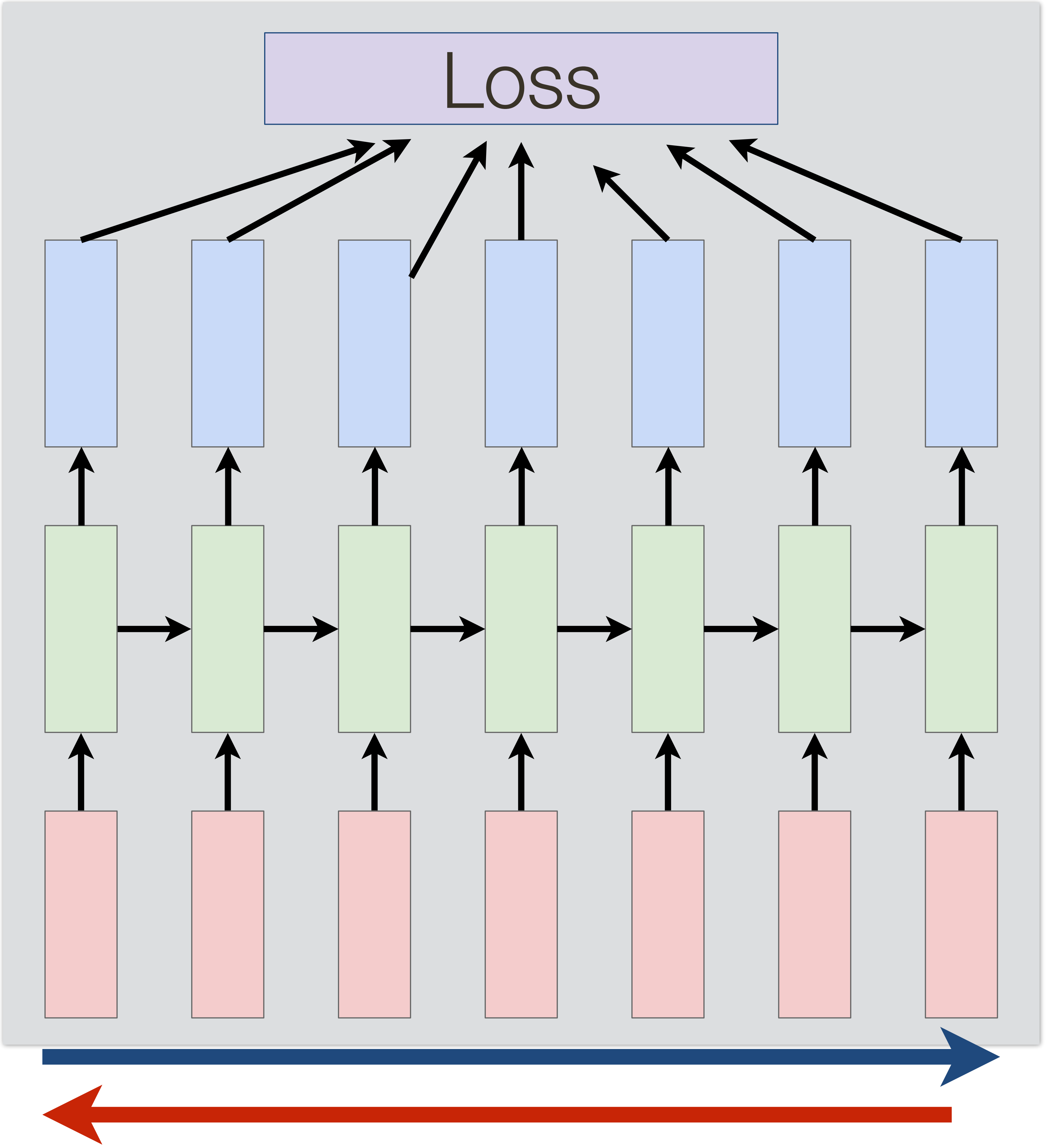
Forward through entire sequence to compute loss, then backward through entire sequence to compute gradient



* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

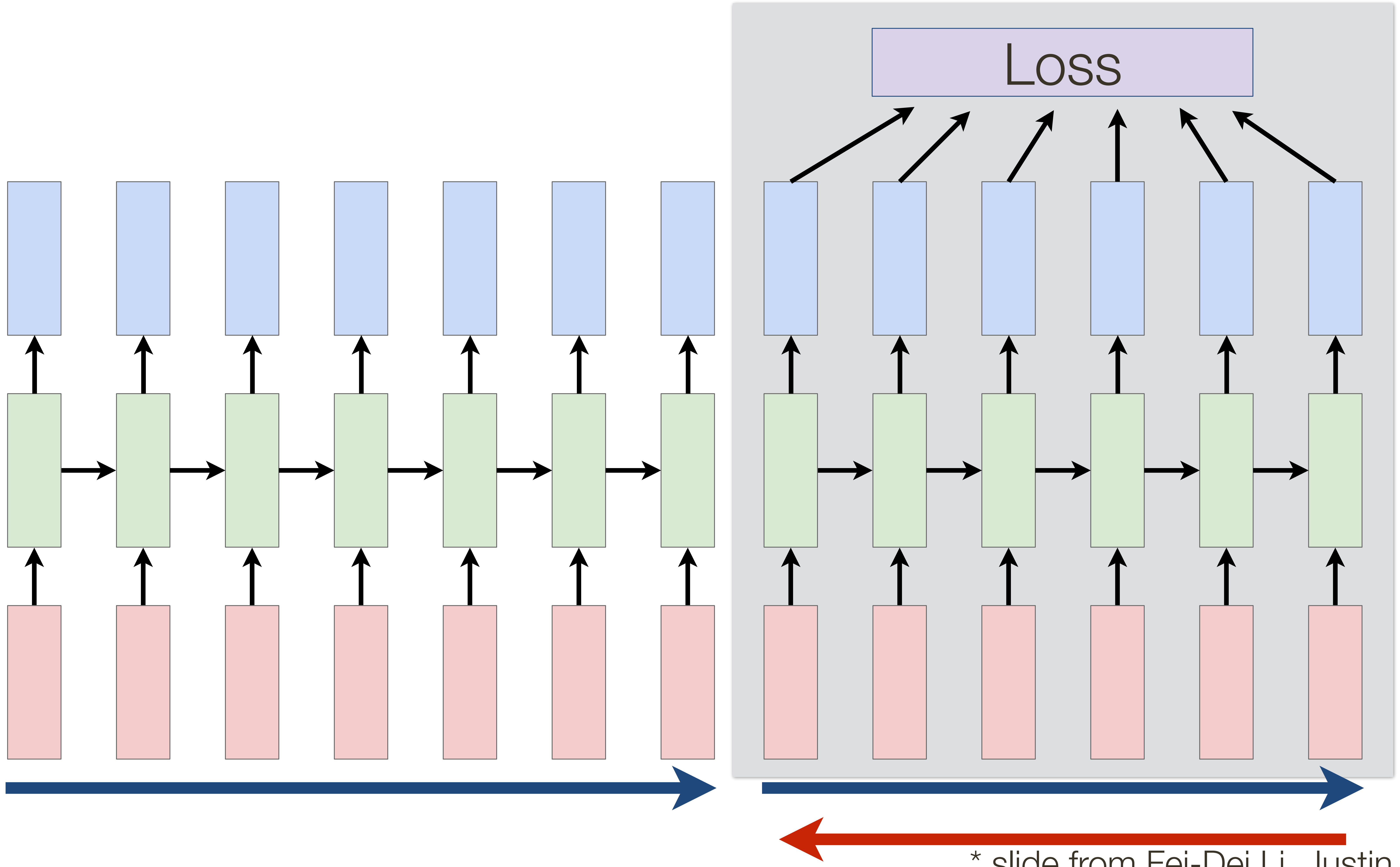
Truncated BackProp Through Time

Run backwards and forwards through (fixed length) **chunks of the sequence**, instead of the whole sequence



Truncated BackProp Through Time

Run backwards and forwards through (fixed length) **chunks of the sequence**, instead of the whole sequence

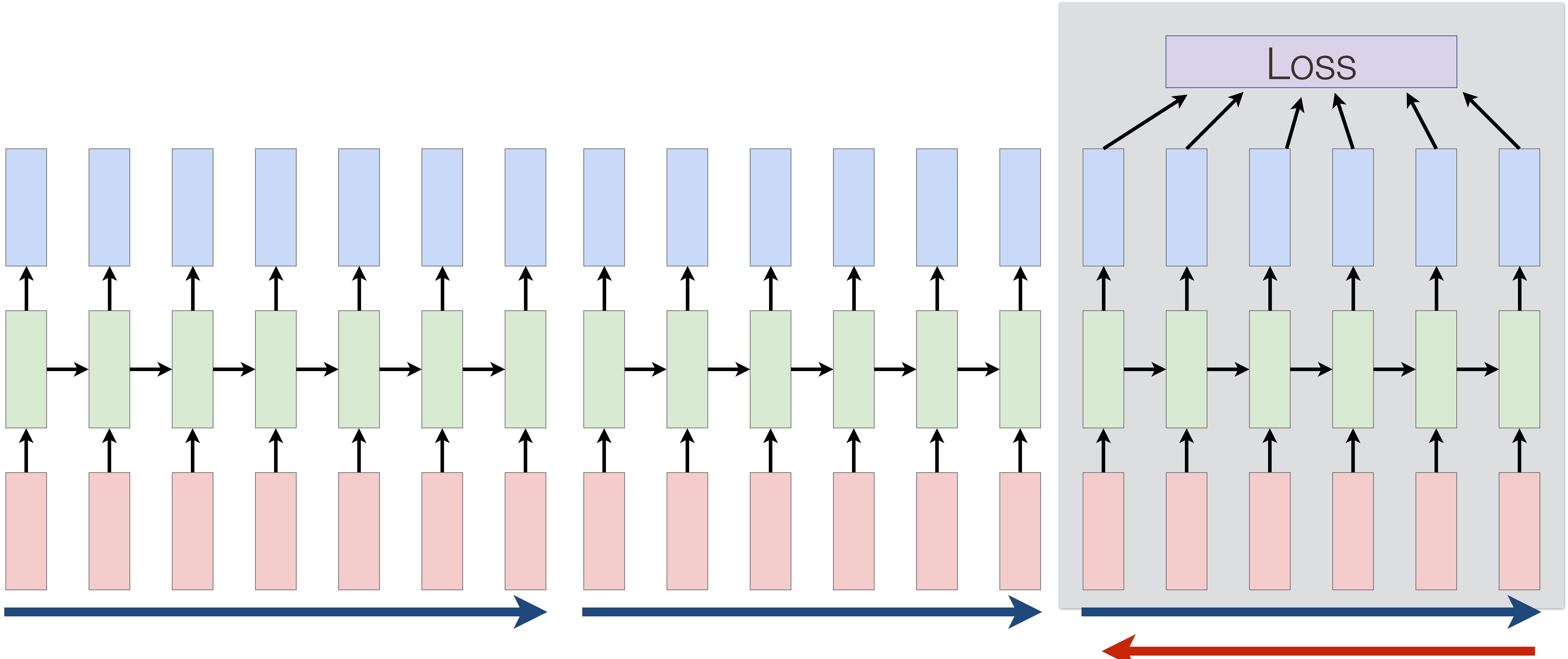


Carry hidden states forward, but only BackProp through some smaller number of steps

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Truncated BackProp Through Time

Run backwards and forwards through (fixed length) **chunks of the sequence**, instead of the whole sequence



* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Implementation: Relatively Easy

… you will have a chance to experience this in the **Assignment 3**

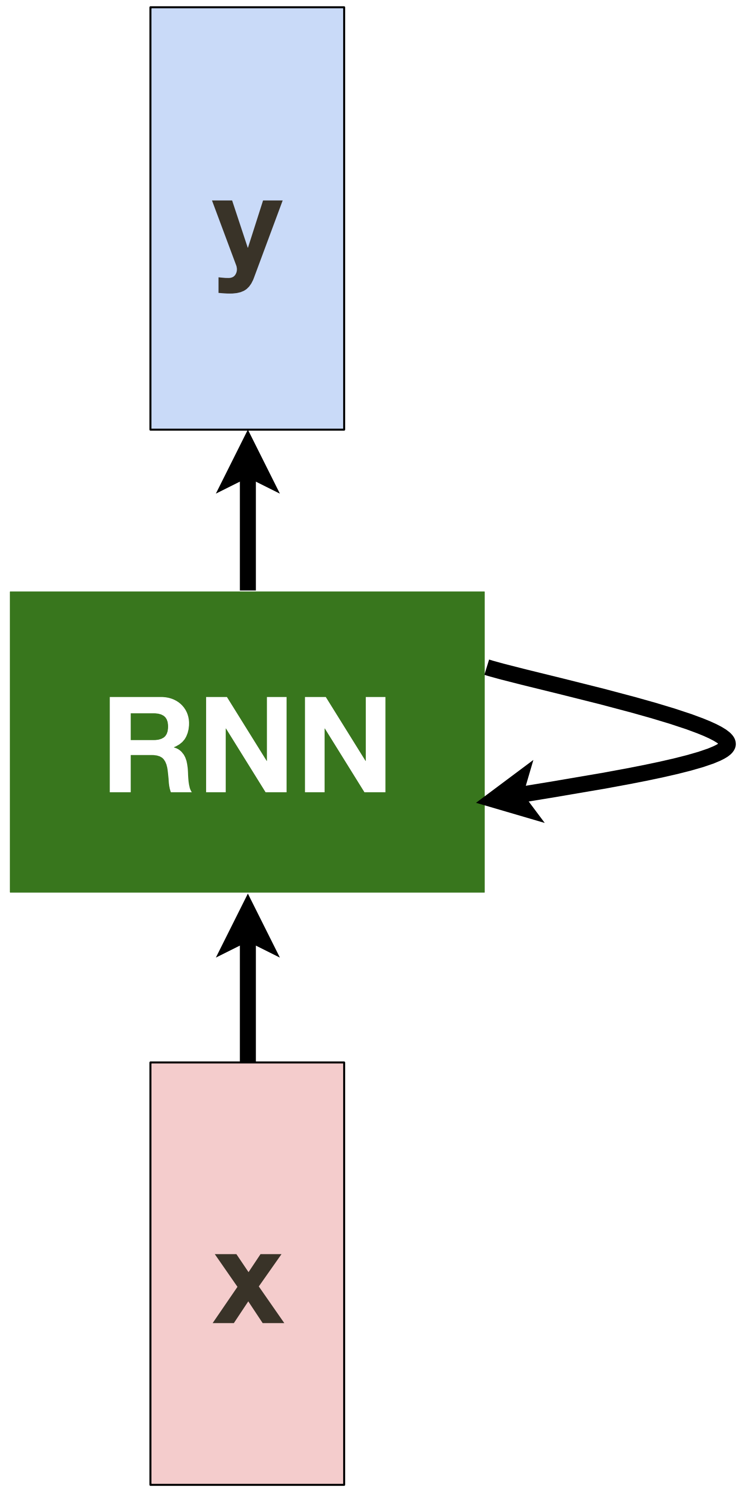
Learning to Write Like Shakespeare

THE SONNETS

by William Shakespeare

From fairest creatures we desire increase,
That thereby beauty's rose might never die,
But as the riper should by time decease,
His tender heir might bear his memory:
But thou, contracted to thine own bright eyes,
Feed'st thy light's flame with self-substantial fuel,
Making a famine where abundance lies,
Thyself thy foe, to thy sweet self too cruel:
Thou that art now the world's fresh ornament,
And only herald to the gaudy spring,
Within thine own bud buriest thy content,
And tender churl mak'st waste in niggarding:
    Pity the world, or else this glutton be,
    To eat the world's due, by the grave and thee.

When forty winters shall besiege thy brow,
And dig deep trenches in thy beauty's field,
Thy youth's proud livery so gazed on now,
Will be a tatter'd weed of small worth held:
Then being asked, where all thy beauty lies,
Where all the treasure of thy lusty days;
To say, within thine own deep sunken eyes,
Were an all-eating shame, and thriftless praise.
How much more praise deserv'd thy beauty's use,
If thou couldst answer 'This fair child of mine
Shall sum my count, and make my old excuse,'
Proving his beauty by succession thine!
    This were to be new made when thou art old,
    And see thy blood warm when thou feel'st it cold.



* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Learning to Write Like Shakespeare … after training a bit

at first:

```
tyntd-iafhatawiaoihrdemot  lytdws  e ,tfti, astai f ogoh eoase rrranbyne 'nhthnee e
plia tklrgd t o idoe ns,smtt   h ne etie h,hregtrs nigtike,aoaenns lng
```

↓ train more

```
"Tmont thithey" fomesscerliund
Keushey. Thom here
sheulke, anmerenith ol sivh I lalterthend Bleipile shuwy fil on aseterlome
coaniogennc Phe lism thond hon at. MeiDimorotion in ther thize."
```

↓ train more

```
Aftair fall unsuch that the hall for Prince Velzonski's that me of
her hearly, and behs to so arwage fiving were to it beloge, pavu say falling misfort
how, and Gogition is so overelical and ofter.
```

↓ train more

```
"Why do what that day," replied Natasha, and wishing to himself the fact the
princess, Princess Mary was easier, fed in had oftened him.
Pierre aking his soul came to the packs and drove up his father-in-law women.
```

Learning to Write Like Shakespeare … after training

```
PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.

DUKE VINCENTIO:
Well, your wit is in the care of side and that.

Second Lord:
They would be ruled after this chamber, and
my fair nues begun out of the fact, to be conveyed,
Whose noble souls I'll have the heart of the wars.

Clown:
Come, sir, I will make did behold your worship.

VIOLA:
I'll drink it.
```

```
VIOLA:
Why, Salisbury must find his flesh and thought
That which I am not aps, not a man and in fire,
To show the reining of the raven and the wars
To grace my hand reproach within, and not a fair are hand,
That Caesar and my goodly father's world;
When I was heaven of presence and our fleets,
We spare with hours, but cut thy council I am great,
Murdered and by thy master's ready there
My power to give thee but so much as hell:
Some service in the noble bondman here,
Would show him to her wine.

KING LEAR:
O, if you were a feeble sight, the courtesy of your law,
Your sight and several breath, will wear the gods
With his heads, and my hands are wonder'd at the deeds,
So drop upon your lordship's head, and your opinion
Shall be against your honour.
```

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Learning Code

Trained on entire source code of Linux kernel

```
static void do_command(struct seq_file *m, void *v)
{
  int column = 32 << (cmd[2] & 0x80);
  if (state)
    cmd = (int)(int_state ^ (in_8(&ch->ch_flags) & Cmd) ? 2 : 1);
  else
    seq = 1;
  for (i = 0; i < 16; i++) {
    if (k & (1 << 1))
      pipe = (in_use & UMXTHREAD_UNCCA) +
        ((count & 0x00000000fffffff8) & 0x000000f) << 8;
    if (count == 0)
      sub(pid, ppc_md.kexec_handle, 0x20000000);
    pipe_set_bytes(i, 0);
  }
  /* Free our user pages pointer to place camera if all dash */
  subsystem_info = &of_changes[PAGE_SIZE];
  rek_controls(offset, idx, &soffset);
  /* Now we want to deliberately put it to device */
  control_check_polarity(&context, val, 0);
  for (i = 0; i < COUNTER; i++)
    seq_puts(s, "policy ");
}
```

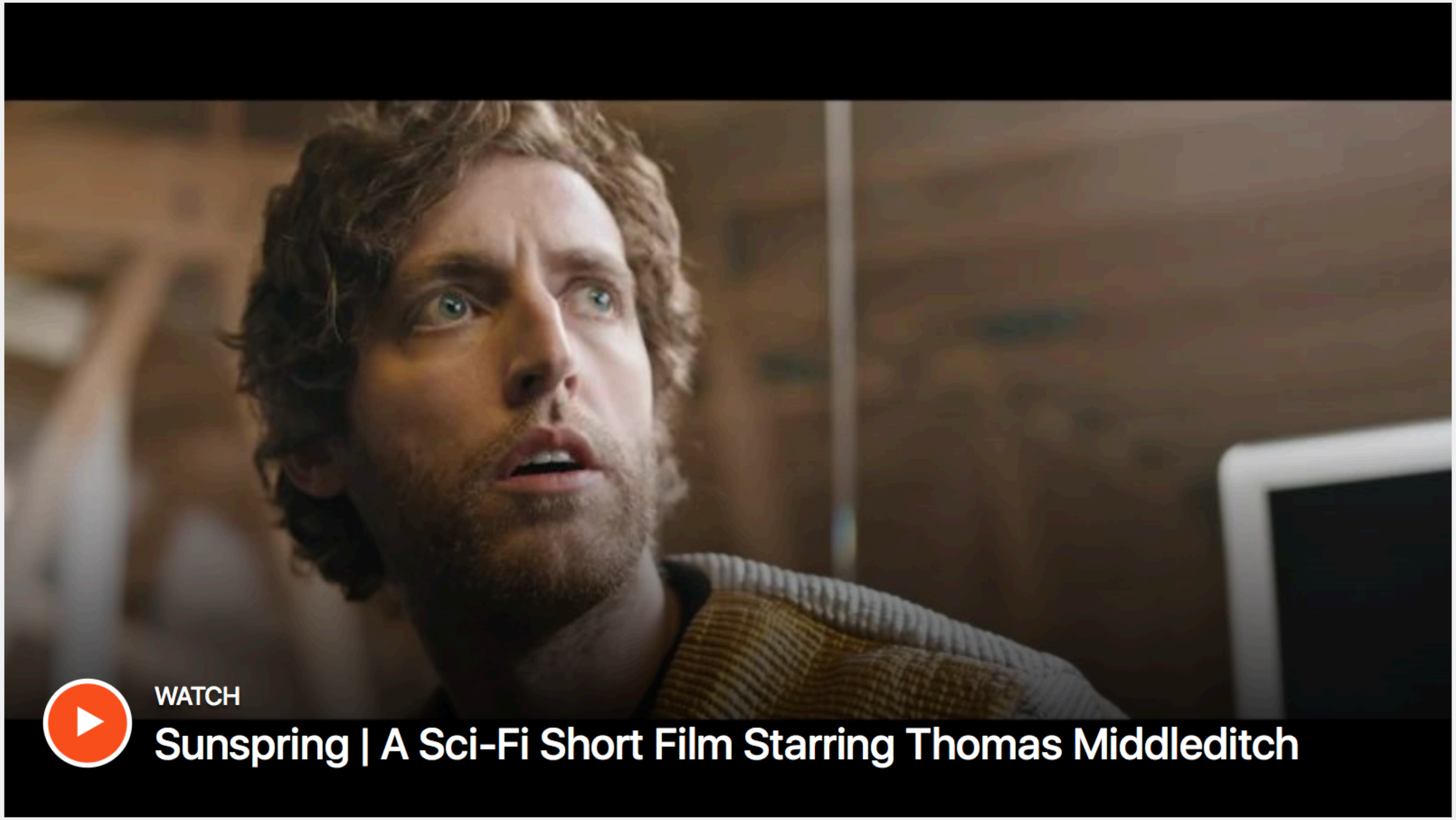
* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

DopeLearning: Computational Approach to Rap Lyrics

Everybody got one	(2 Chainz - Extremely Blessed)
And all the pretty mommies want some	(Mos Def - Undeniable)
And what i told you all was	(Lil Wayne - Welcome Back)
But you need to stay such do not touch	(Common - Heidi Hoe)
They really do not want you to vote	(KRS One - The Mind)
what do you condone	(Cam'ron - Bubble Music)
Music make you lose control	(Missy Elliot - Lose Control)
What you need is right here ahh oh	(Wiz Khalifa - Right Here)
This is for you and me	(Missy Elliot - Hit Em Wit Da Hee)
I had to dedicate this song to you Mami	(Fat Joe - Bendicion Mami)
Now I see how you can be	(Lil Wayne - How To Hate)
I see u smiling i kno u hattig	(Wiz Khalifa - Damn Thing)
Best I Eva Had x4	(Nicki Minaj - Best I Ever Had)
That I had to pay for	(Ice Cube - X Bitches)
Do I have the right to take yours	(Common - Retrospect For Life)
Trying to stay warm	(Everlast - 2 Pieces Of Drama)

[Malmi et al., KDD 2016]

Sunspring: First movie generated by AI

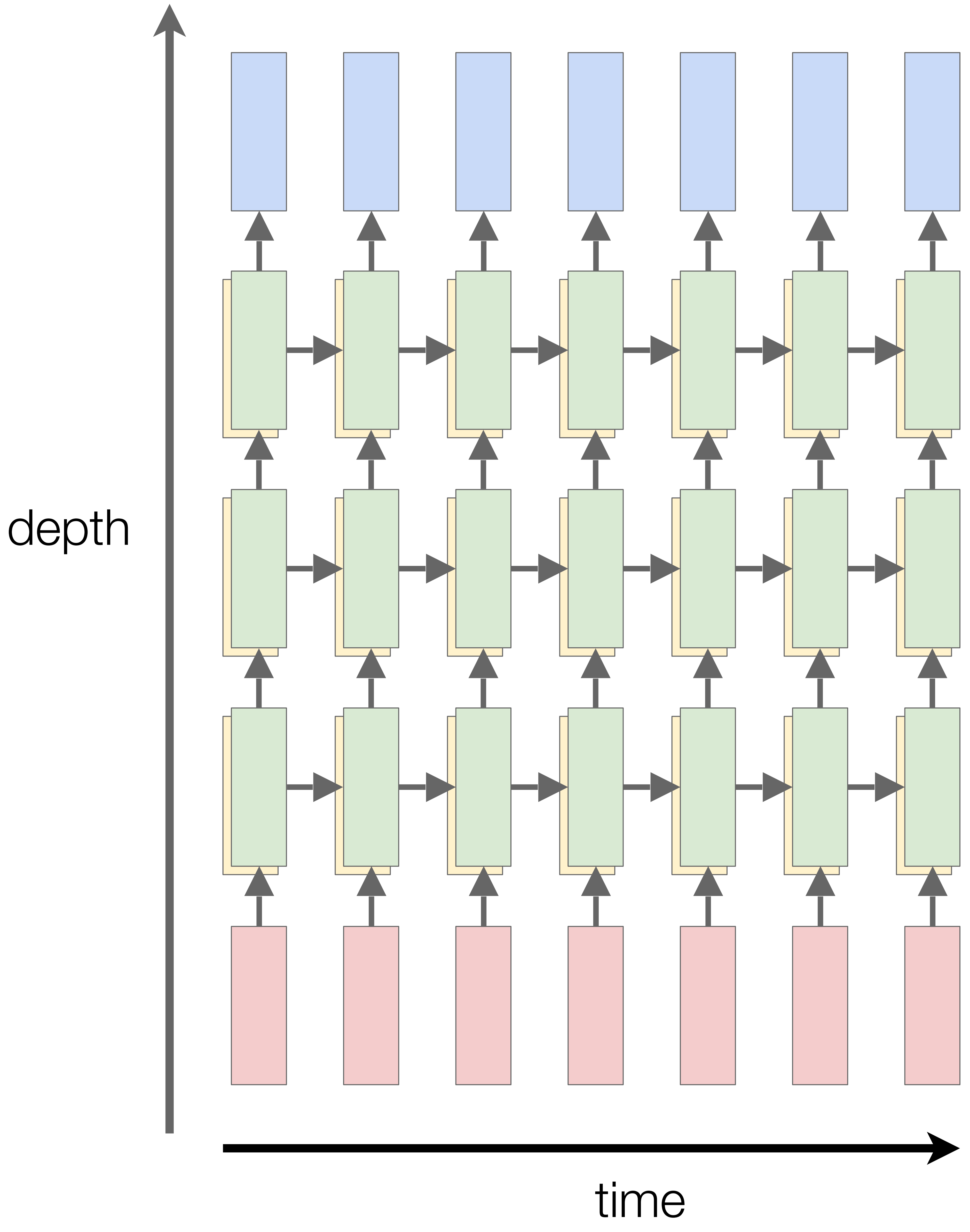


Sunspring, a short science fiction movie written entirely by AI, debuts exclusively on Ars today.

Multilayer RNNs

$$\boxed{h_t^l = \tanh W^l \begin{pmatrix} h_t^{l-1} \\ h_{t-1}^l \end{pmatrix}}$$

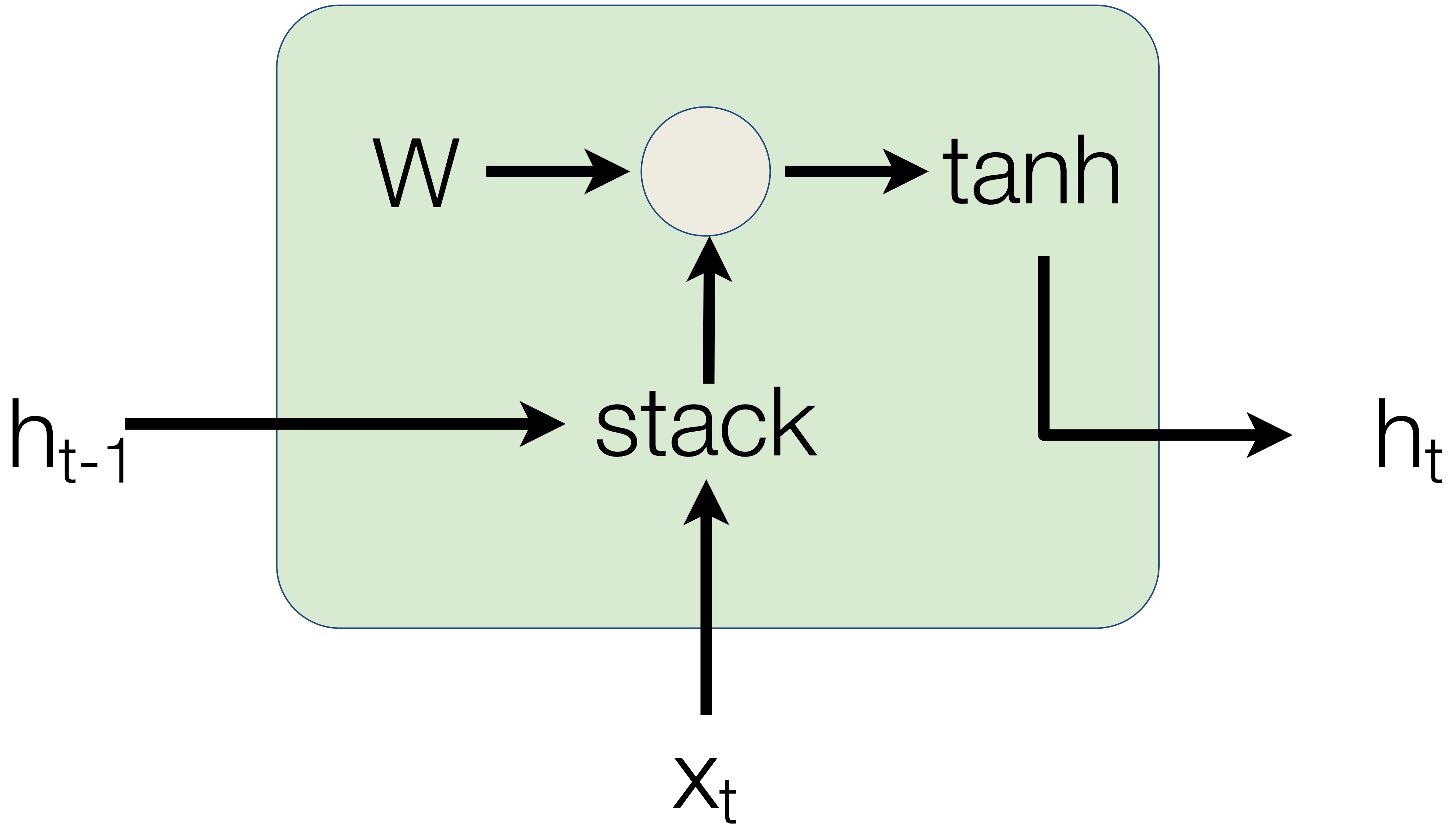
$h \in \mathbb{R}^n$ $\quad\quad$ $W^l \;\; [n \times 2n]$



* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Vanilla RNN **Gradient Flow**

[Bengio et al., 1994]
[Pascanu et al., ICML 2013]

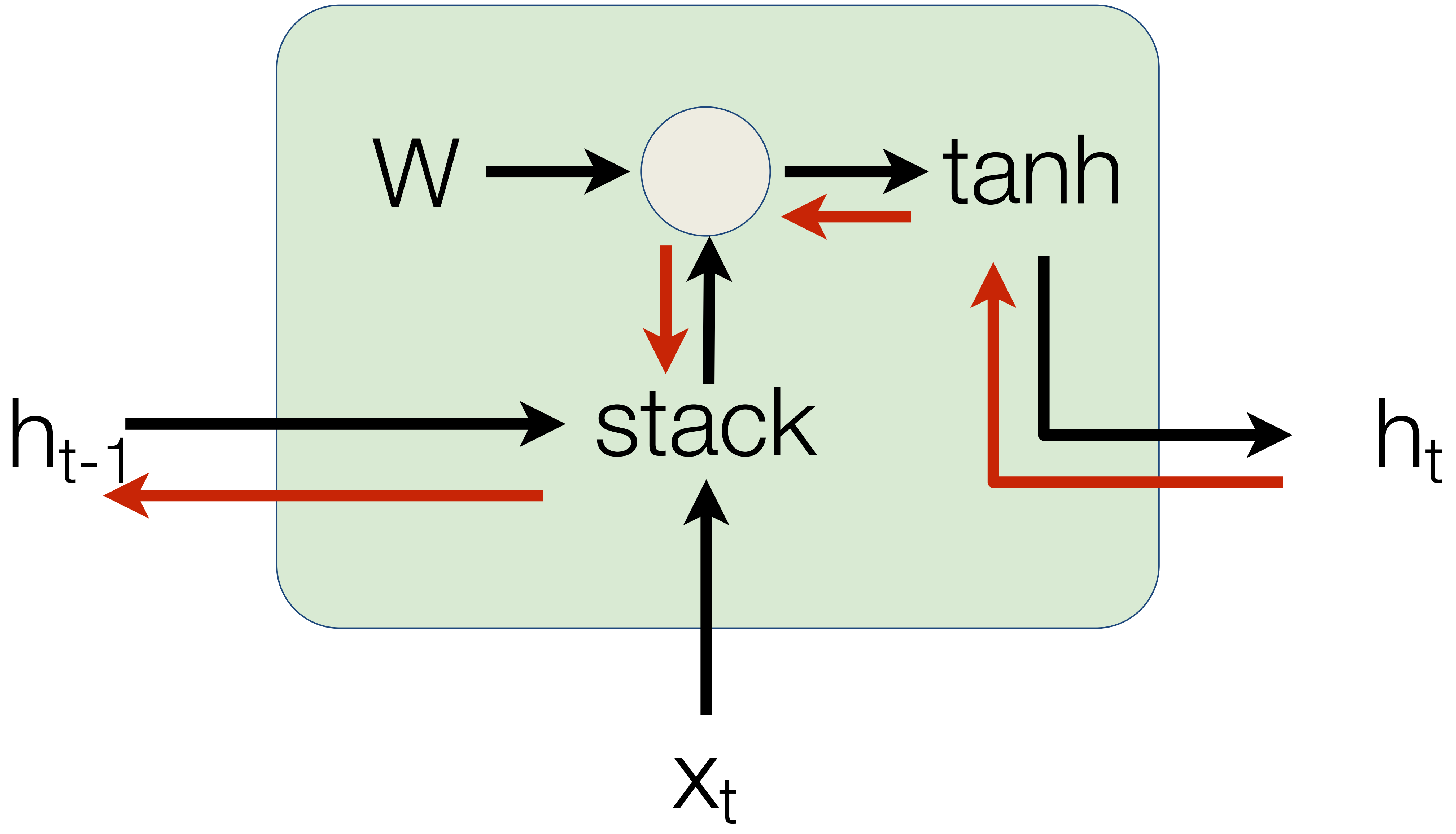


$$
\begin{aligned}
h_t &= \tanh(W_{hh}h_{t-1} + W_{xh}x_t) \\
&= \tanh\left(\begin{pmatrix} W_{hh} & W_{hx} \end{pmatrix} \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right) \\
&= \tanh\left(W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)
\end{aligned}
$$

Vanilla RNN **Gradient Flow**

[Bengio et al., 1994]
[Pascanu et al., ICML 2013]

Backpropagation from h_t to h_{t-1} multiplies by W (actually W_{hh}^T)

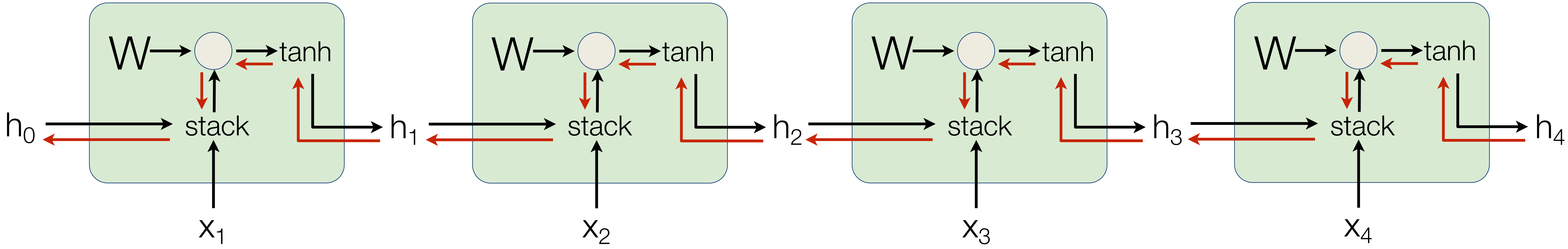


$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$
$$= \tanh\left(\begin{pmatrix} W_{hh} & W_{hx} \end{pmatrix} \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$
$$= \tanh\left(W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

Vanilla RNN **Gradient Flow**

[Bengio et al., 1994]

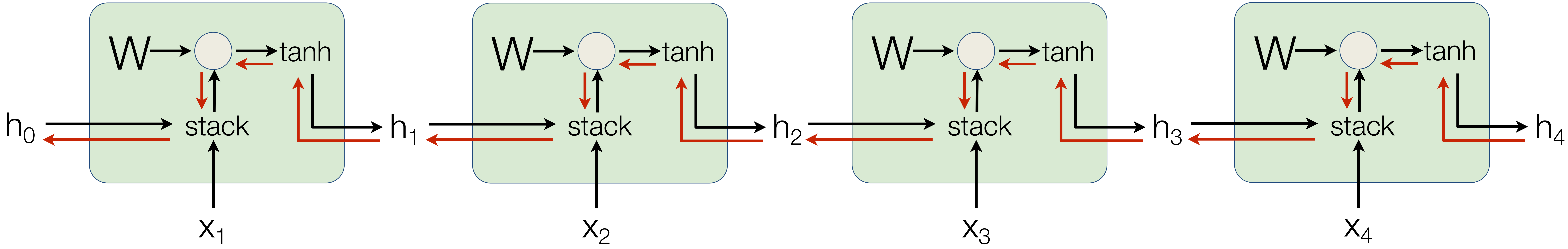
[Pascanu et al., ICML 2013]



Computing gradient of h_0 involves many factors of W (and repeated tanh)

Vanilla RNN **Gradient Flow**

[Bengio et al., 1994]
[Pascanu et al., ICML 2013]



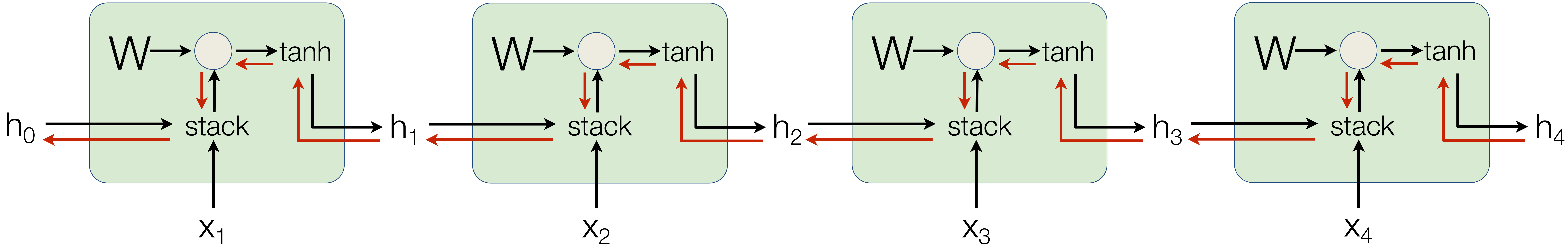
Computing gradient of h_0 involves many factors of W (and repeated tanh)

Largest singular value > 1:
Exploding gradients

Largest singular value < 1:
Vanishing gradients

Vanilla RNN **Gradient Flow**

[Bengio et al., 1994]
[Pascanu et al., ICML 2013]



Computing gradient of h_0 involves many factors of W (and repeated tanh)

Largest singular value > 1:
Exploding gradients

Largest singular value < 1:
Vanishing gradients

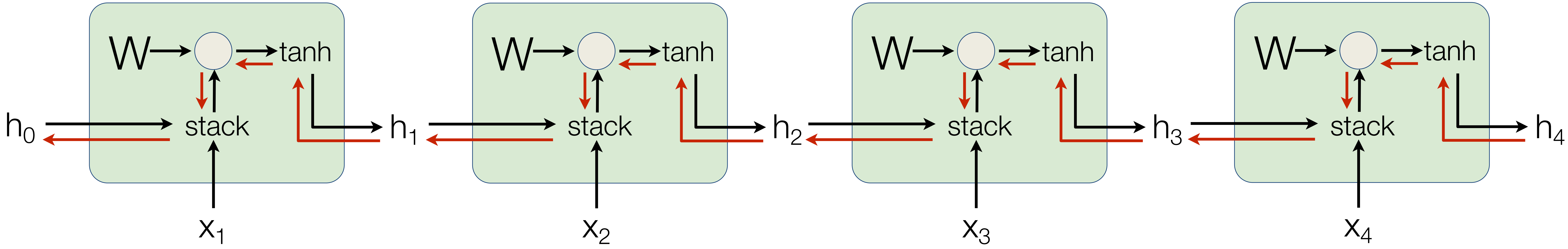
Gradient clipping: Scale gradient if its norm is too big

```
grad_norm = np.sum(grad * grad)
if grad_norm > threshold:
  grad *= (threshold / grad_norm)
```

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Vanilla RNN **Gradient Flow**

[Bengio et al., 1994]
[Pascanu et al., ICML 2013]



Computing gradient of h_0 involves many factors of W (and repeated tanh)

Largest singular value > 1:
Exploding gradients

Largest singular value < 1:
Vanishing gradients

Change RNN architecture

Long-Short Term Memory (**LSTM**)

Vanilla RNN

$$h_t = \tanh\left(W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

LSTM

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

[Hochreiter and Schmidhuber, NC **1977**]

Long-Short Term Memory (**LSTM**)

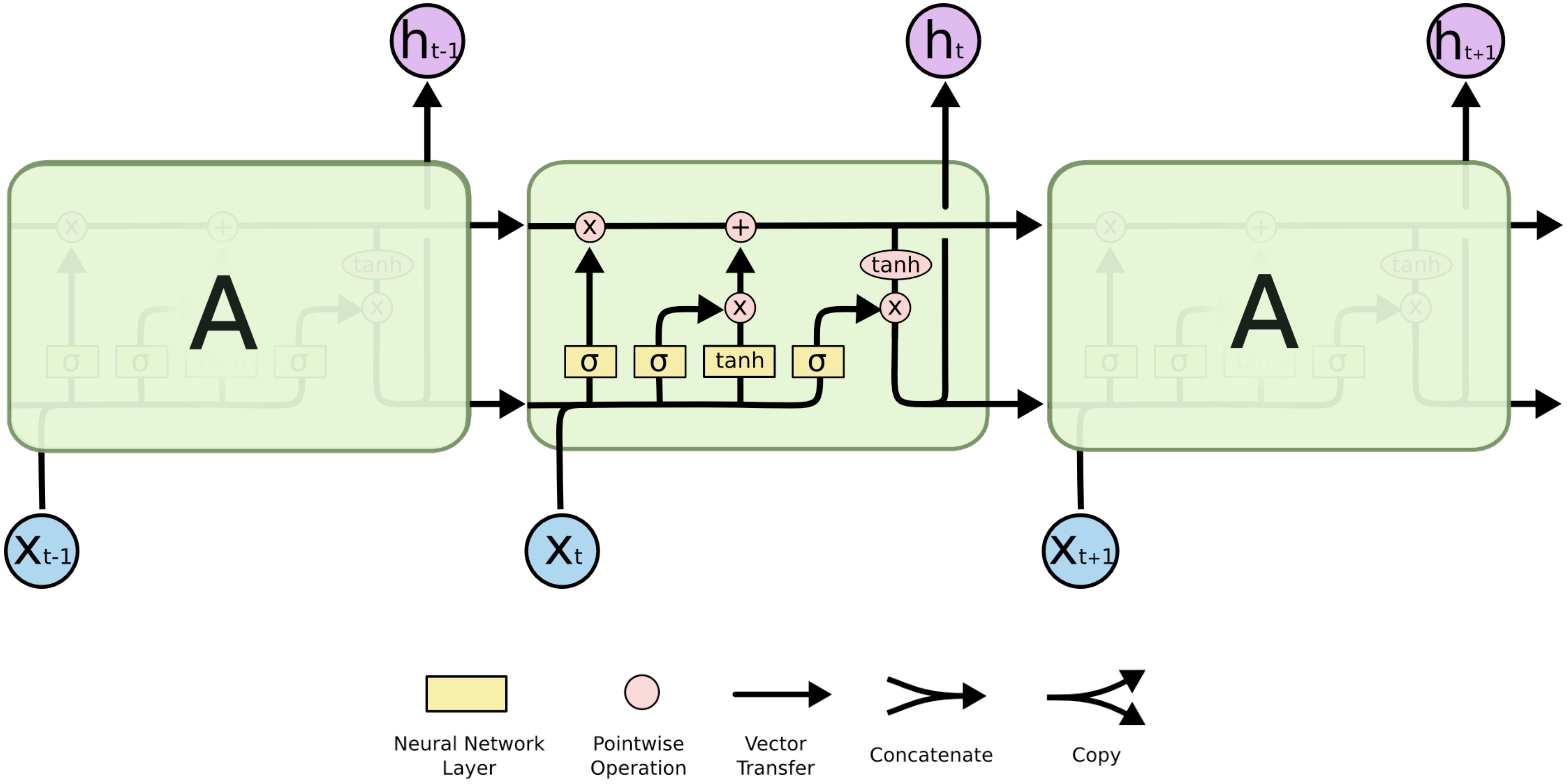


Image Credit: Christopher Olah (http://colah.github.io/posts/2015-08-Understanding-LSTMs/)

* slide from Dhruv Batra

Long-Short Term Memory (**LSTM**)

Cell state / **memory**

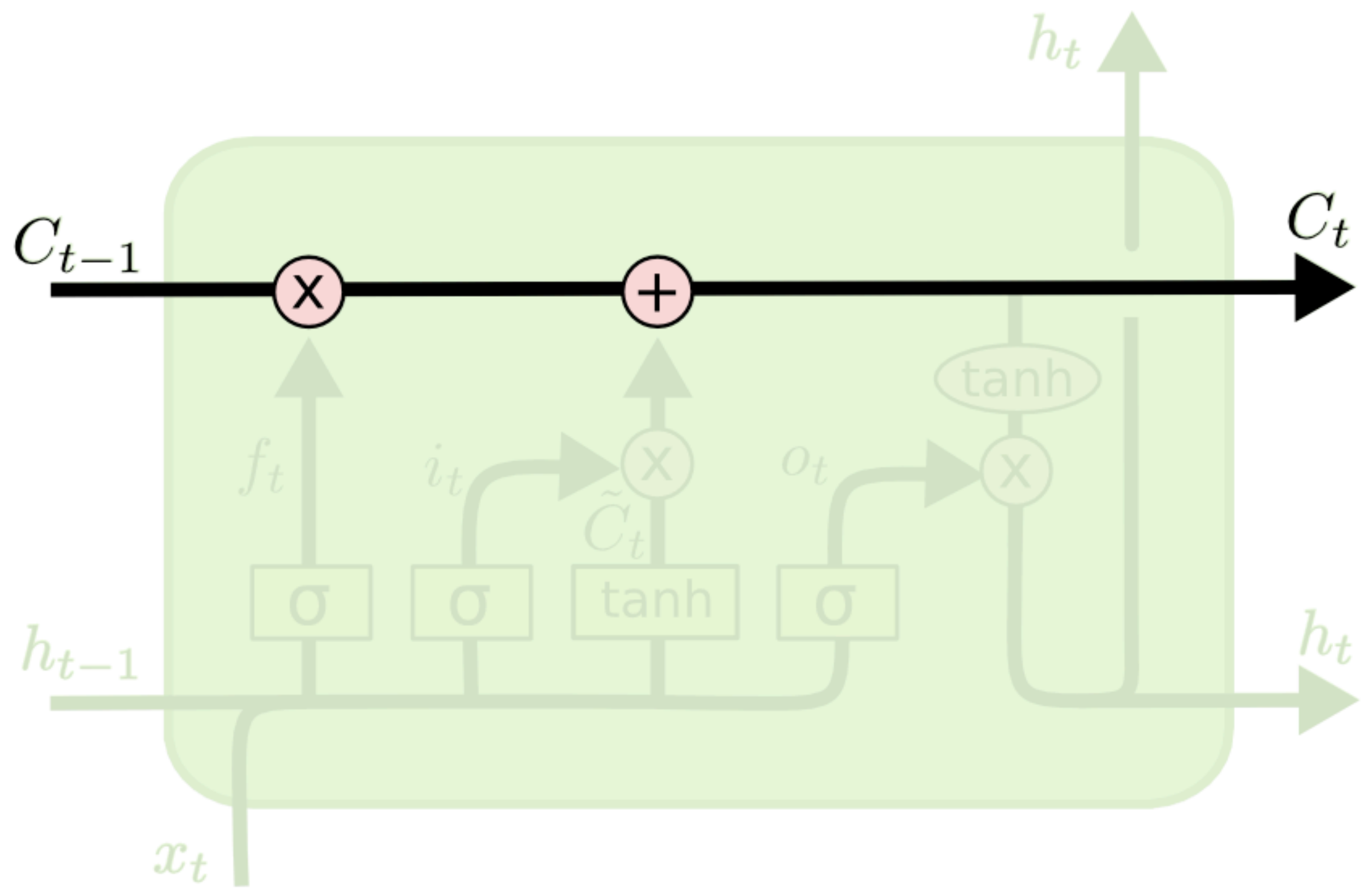
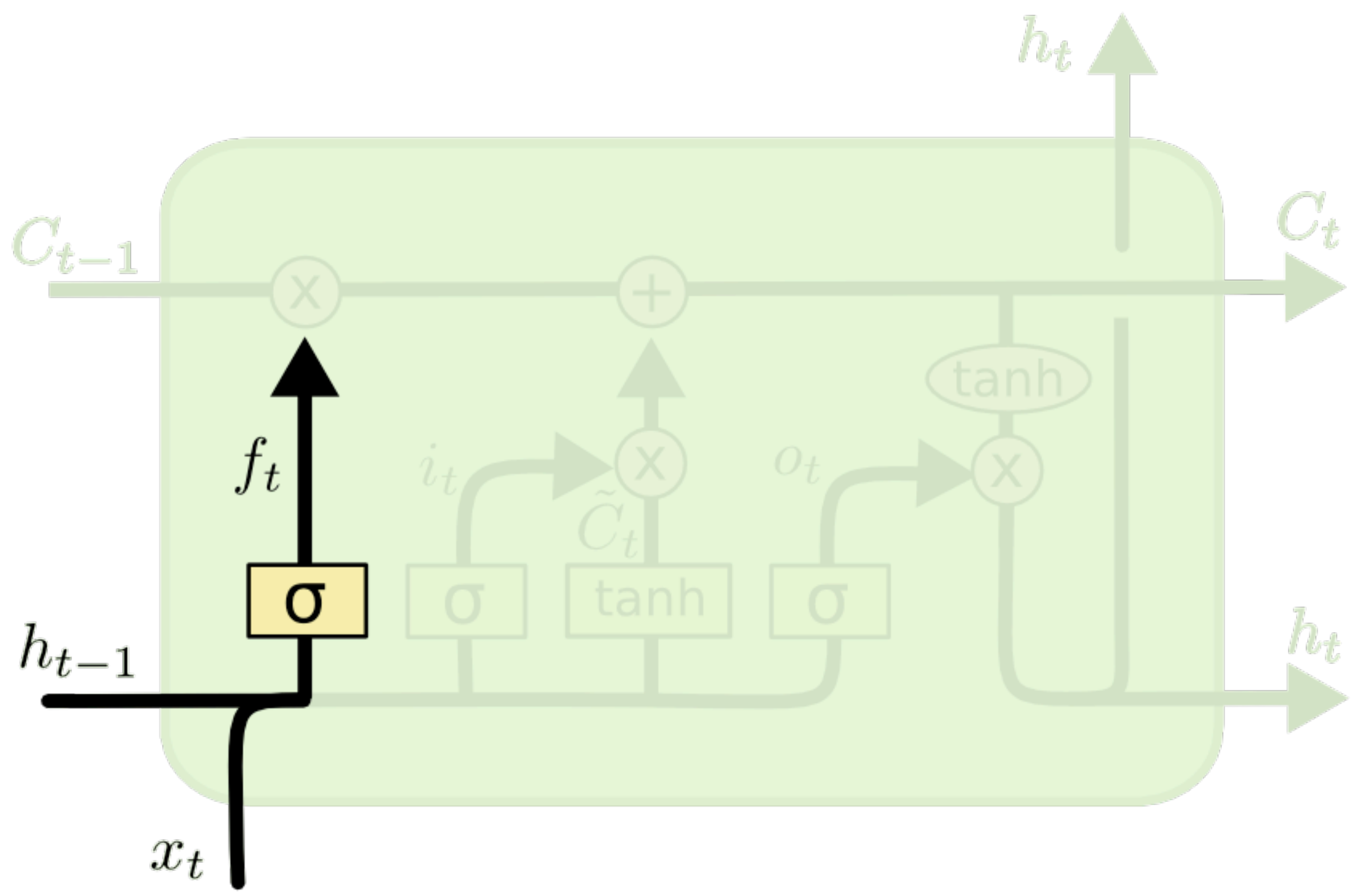


Image Credit: Christopher Olah (http://colah.github.io/posts/2015-08-Understanding-LSTMs/)

* slide from Dhruv Batra

LSTM Intuition: Forget Gate

Should we continue to **remember** this “bit” of information or not?



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

Intuition: memory and forget gate output multiply, output of forget gate can be though of as binary (0 or 1)

anything x 1 = anything (remember)

anything x 0 = 0 (forget)

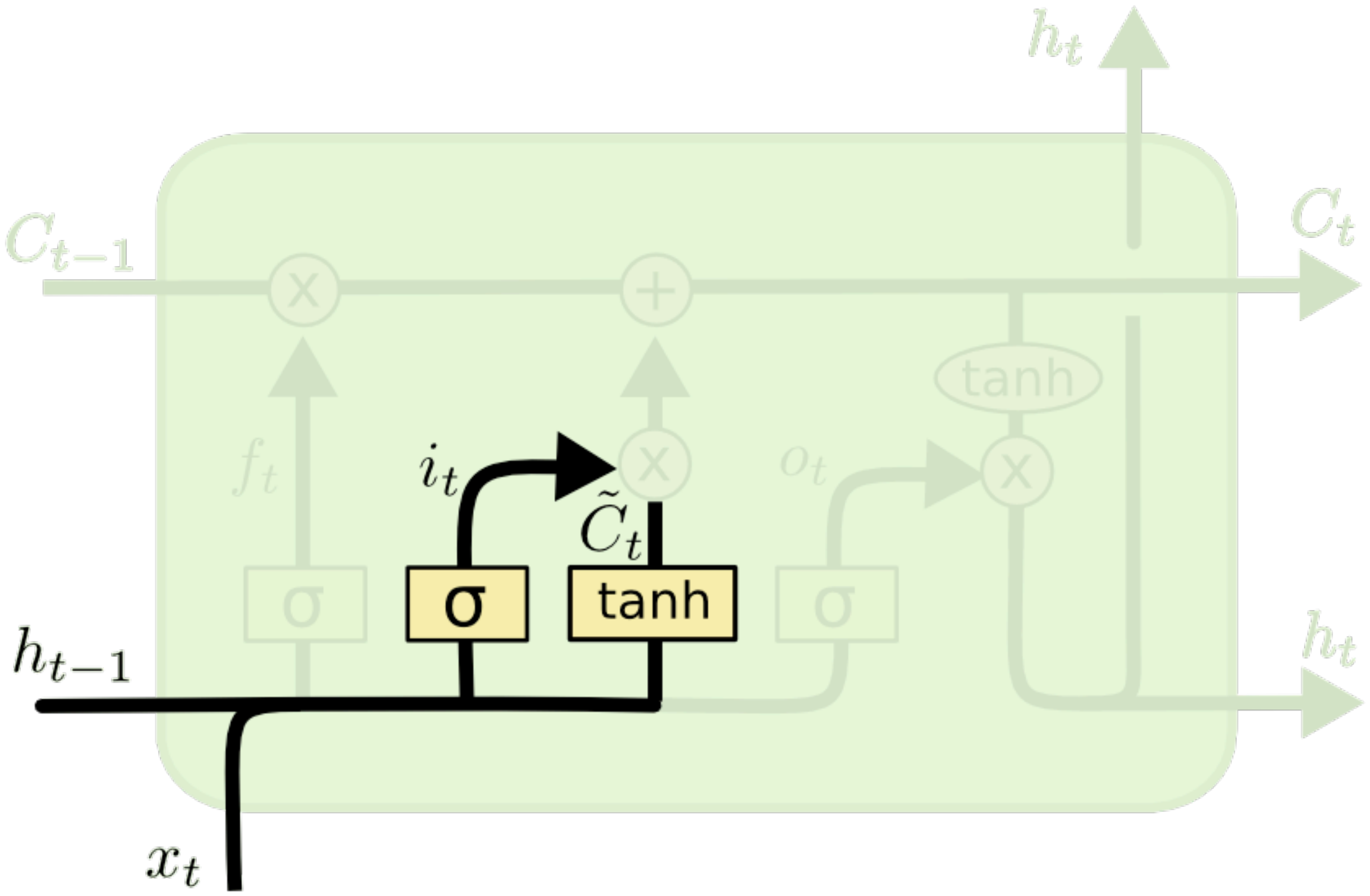
Image Credit: Christopher Olah (http://colah.github.io/posts/2015-08-Understanding-LSTMs/)

* slide from Dhruv Batra

LSTM Intuition: Input Gate

Should we **update** this "bit" of information or not?

If yes, then what should we **remember**?



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

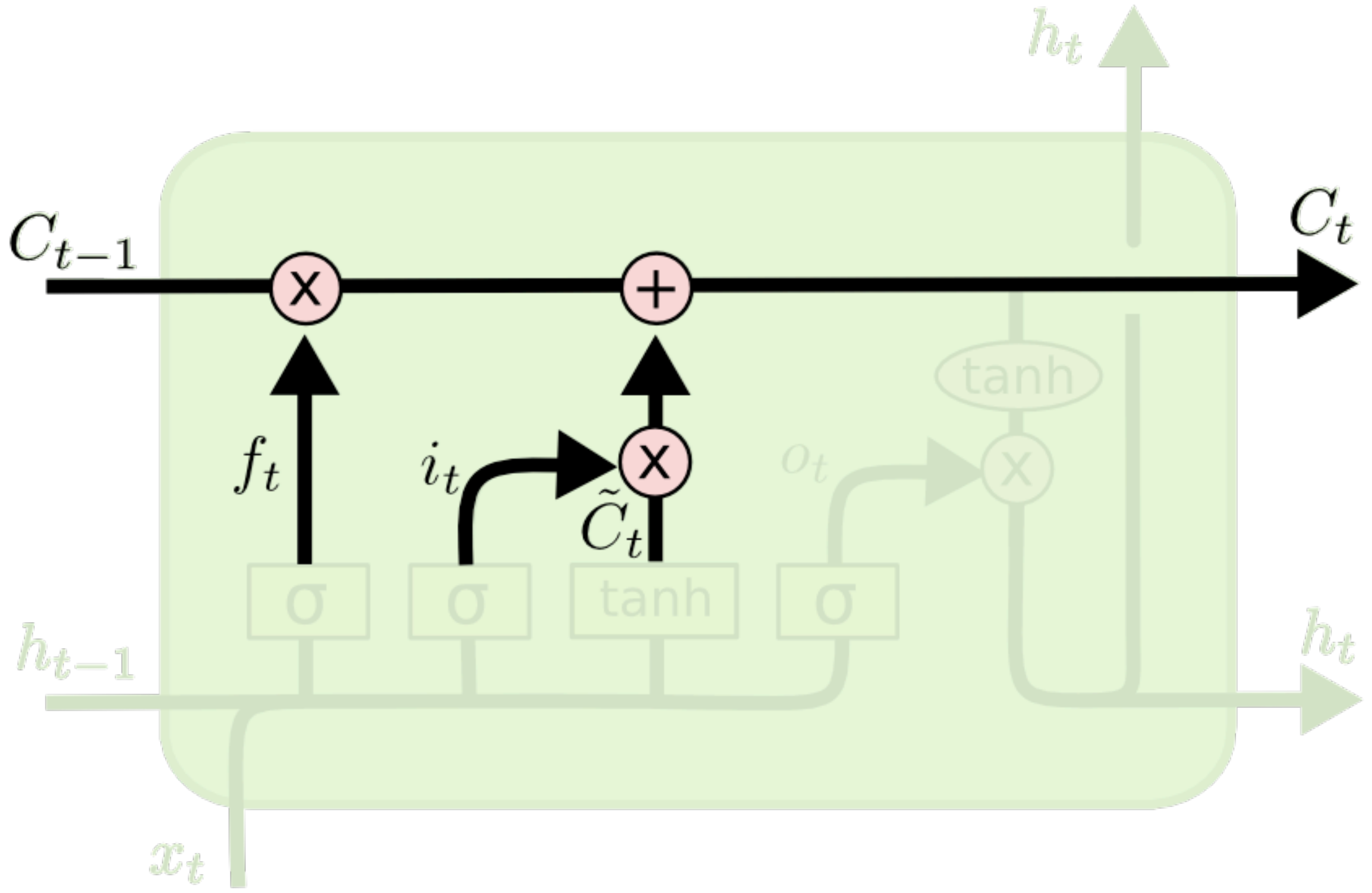
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Image Credit: Christopher Olah (http://colah.github.io/posts/2015-08-Understanding-LSTMs/)

* slide from Dhruv Batra

LSTM Intuition: Memory Update

Forget what needs to be forgotten + memorize what needs to be remembered



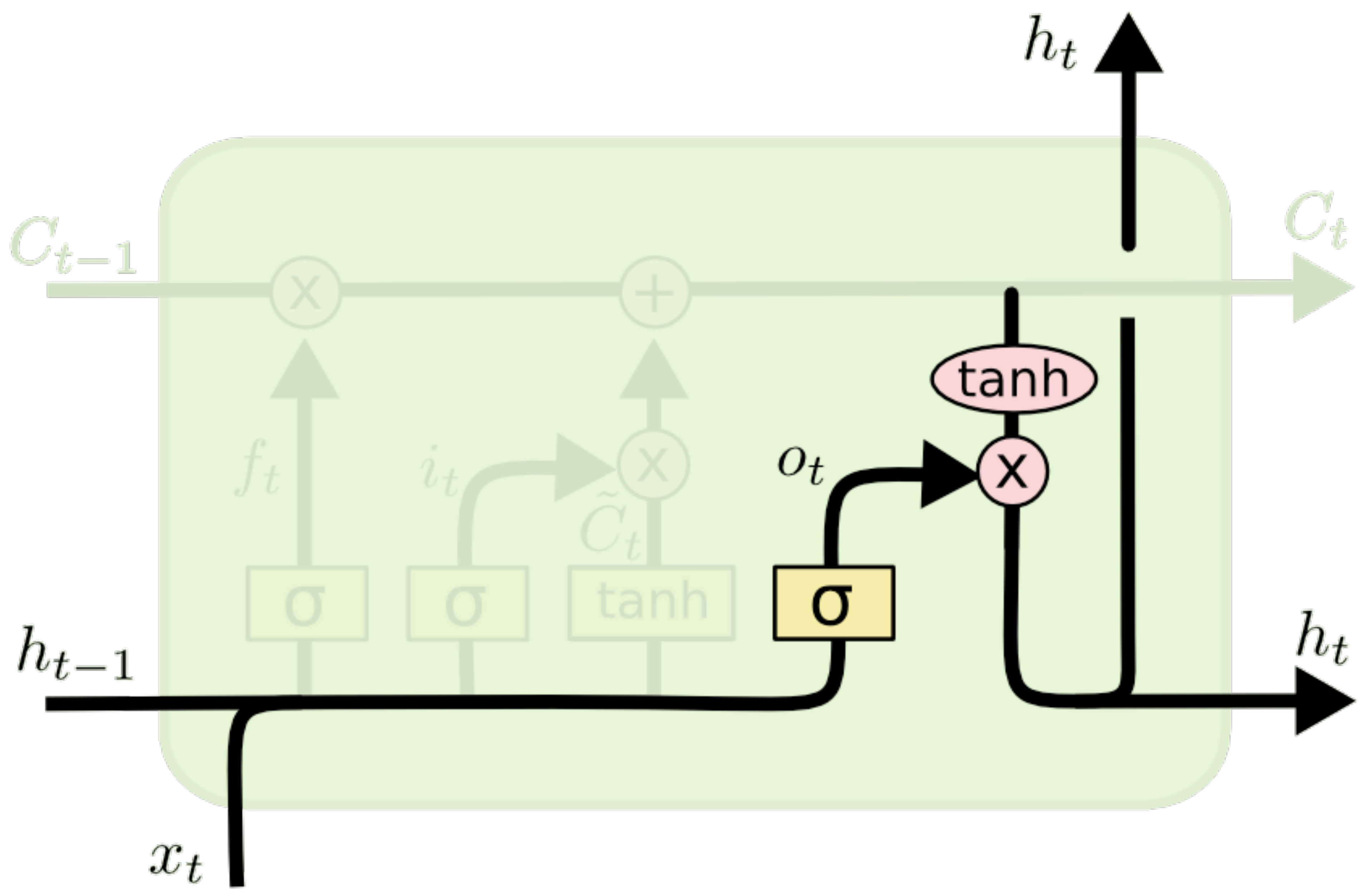
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Image Credit: Christopher Olah (http://colah.github.io/posts/2015-08-Understanding-LSTMs/)

* slide from Dhruv Batra

LSTM Intuition: Output Gate

Should we output this bit of information (e.g., to “deeper” LSTM layers)?



$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

Image Credit: Christopher Olah (http://colah.github.io/posts/2015-08-Understanding-LSTMs/)

* slide from Dhruv Batra

LSTM Intuition: Additive Updates

Backpropagation from c_t to c_{t-1} only elementwise multiplication by f, no matrix multiply by W

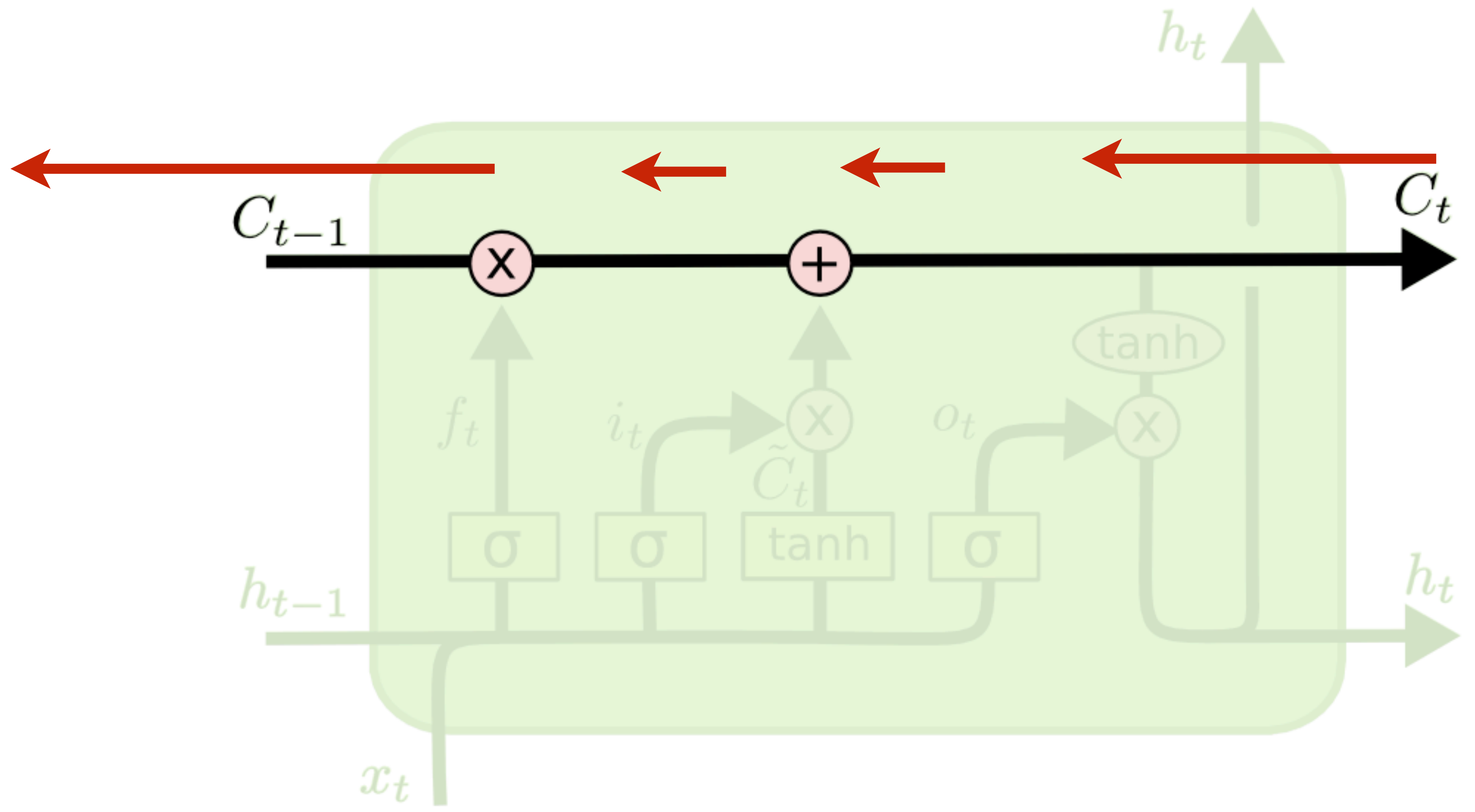
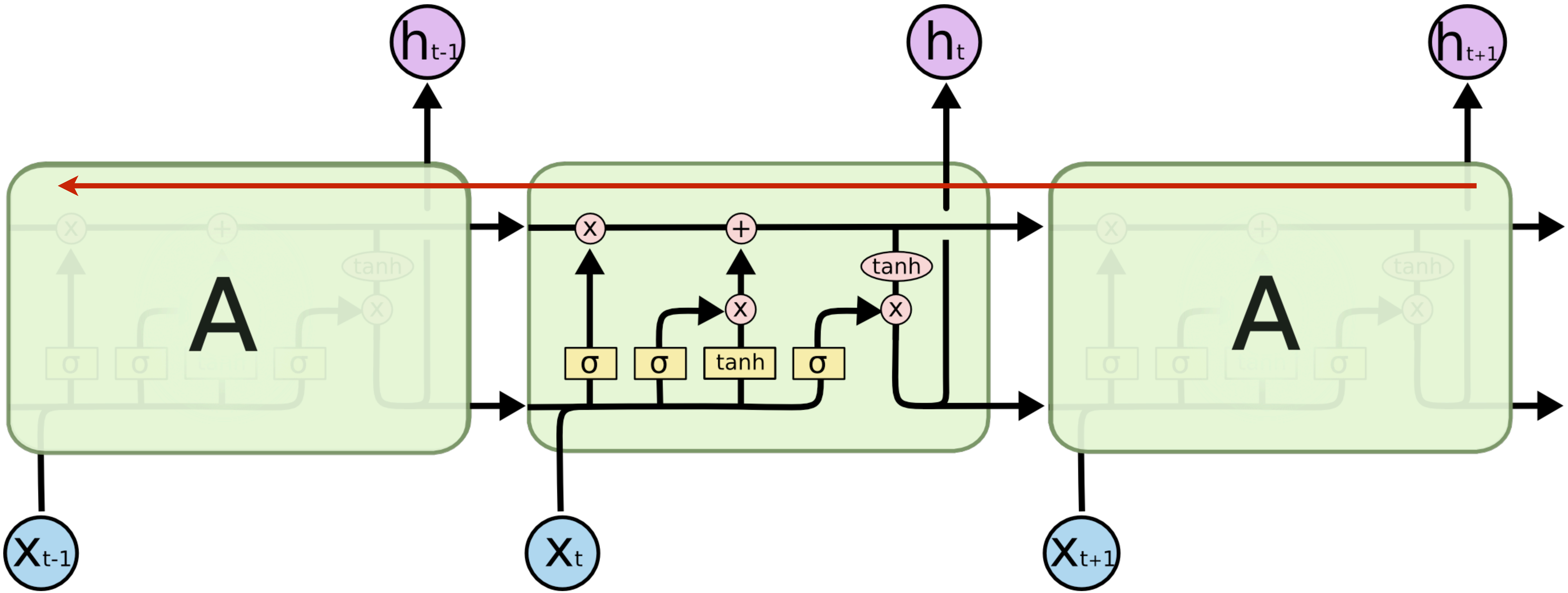


Image Credit: Christopher Olah (http://colah.github.io/posts/2015-08-Understanding-LSTMs/)

* slide from Dhruv Batra

LSTM Intuition: Additive Updates

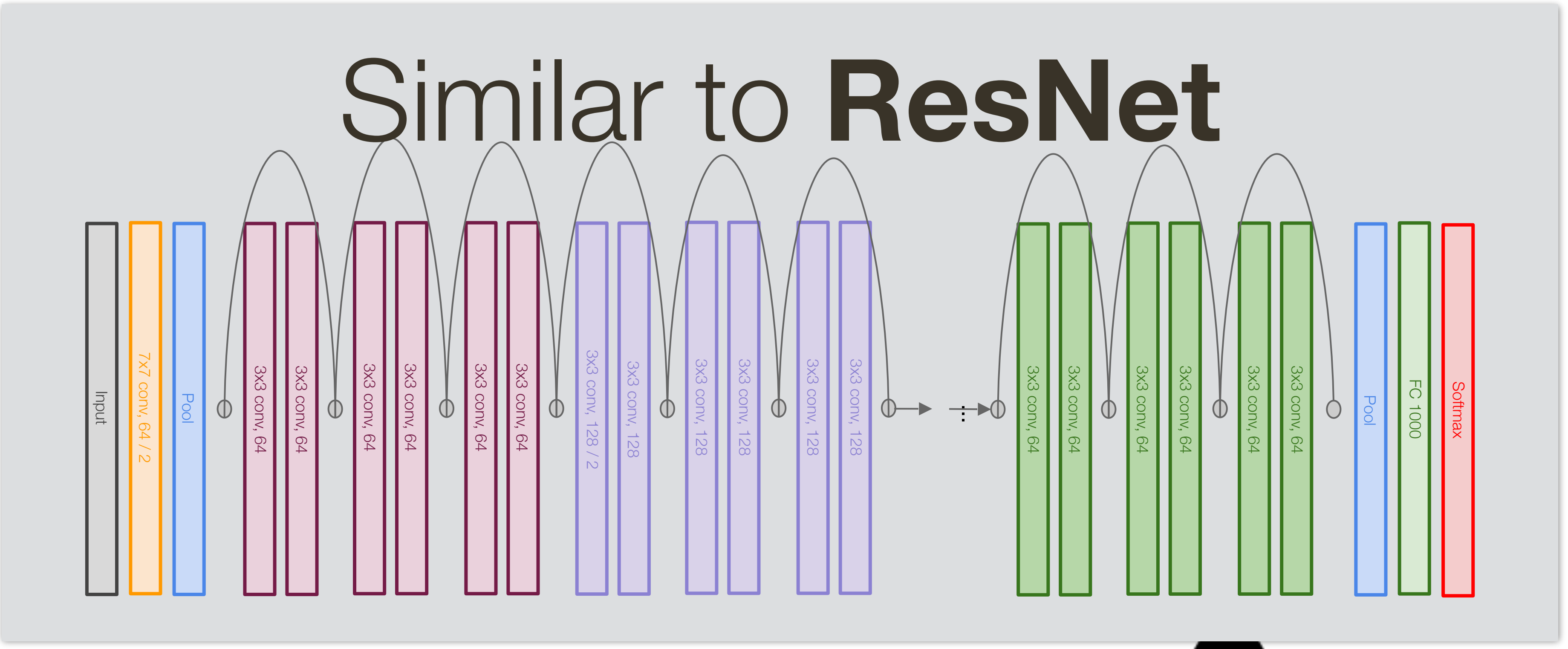


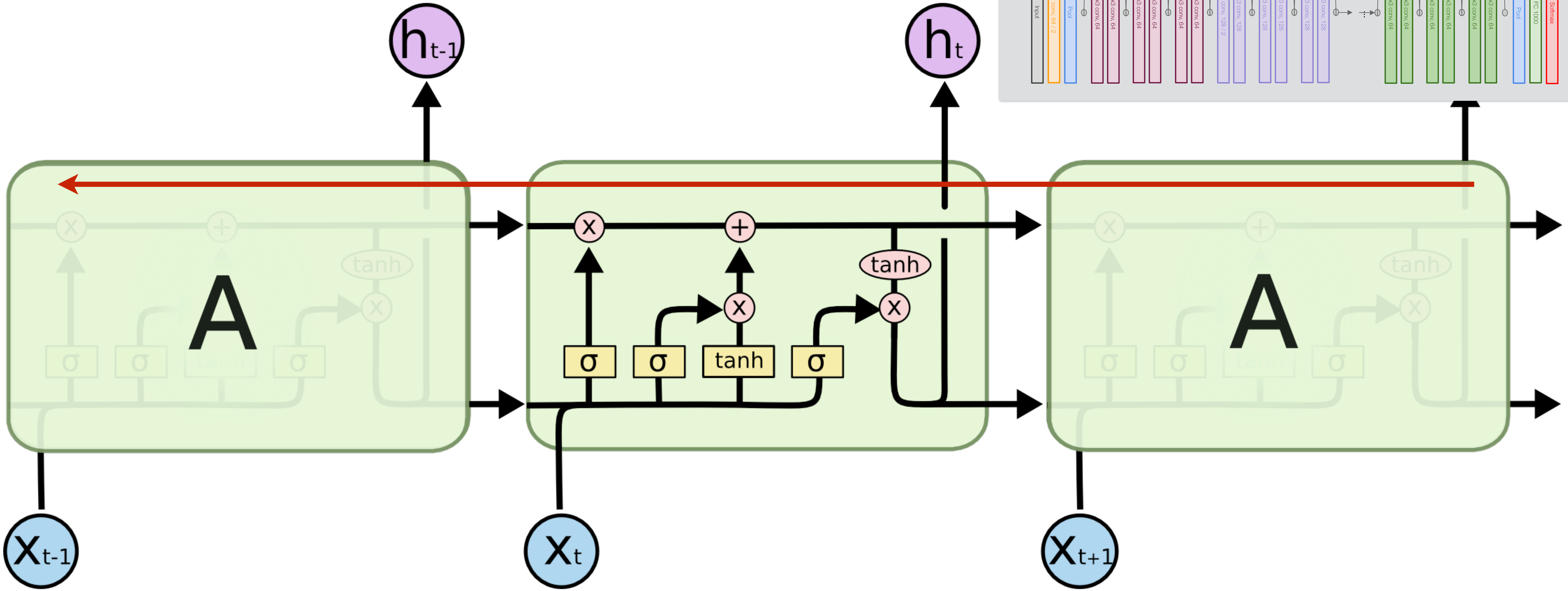
Uninterrupted gradient flow!

Image Credit: Christopher Olah (http://colah.github.io/posts/2015-08-Understanding-LSTMs/)

* slide from Dhruv Batra

LSTM Intuition: Additive Updates





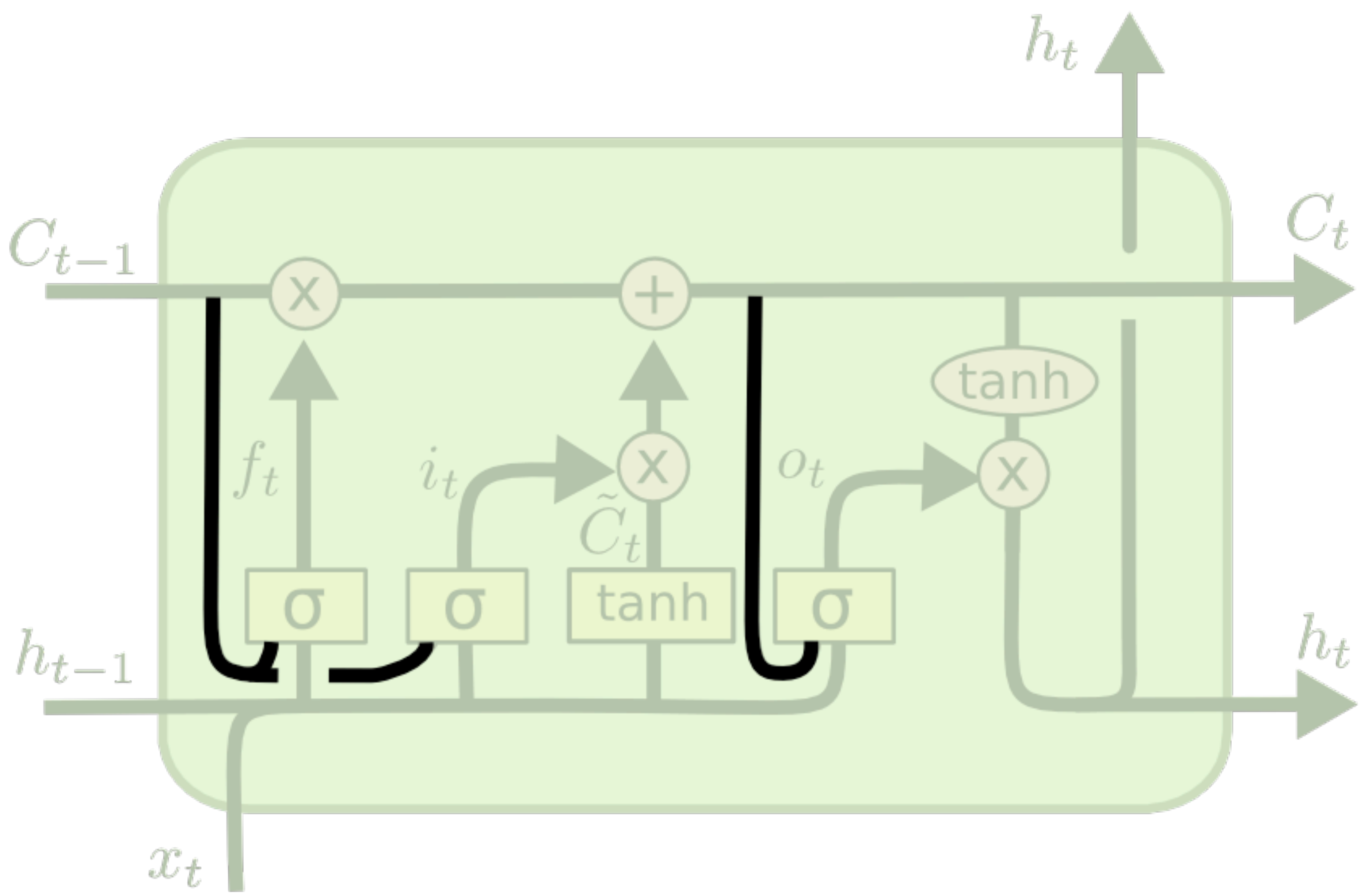
Uninterrupted gradient flow!

Image Credit: Christopher Olah (http://colah.github.io/posts/2015-08-Understanding-LSTMs/)

* slide from Dhruv Batra

LSTM Variants: with Peephole Connections

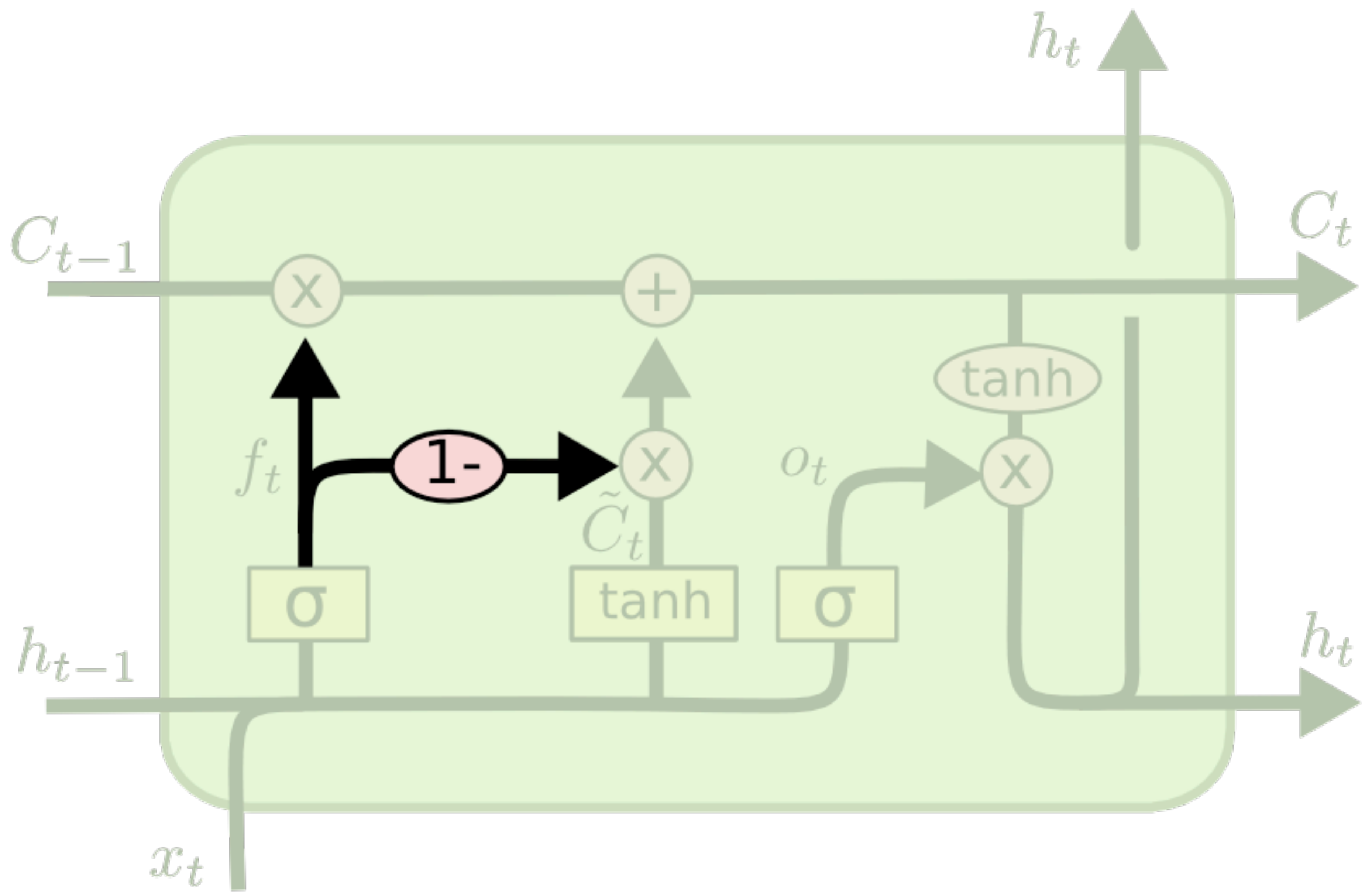
Lets gates see the cell state / memory



$$f_t = \sigma\left(W_f \cdot \left[\boldsymbol{C_{t-1}}, h_{t-1}, x_t\right] \ + \ b_f\right)$$
$$i_t = \sigma\left(W_i \cdot \left[\boldsymbol{C_{t-1}}, h_{t-1}, x_t\right] \ + \ b_i\right)$$
$$o_t = \sigma\left(W_o \cdot \left[\boldsymbol{C_t}, h_{t-1}, x_t\right] \ + \ b_o\right)$$

LSTM Variants: with Coupled Gates

Only memorize new information when you're forgetting old



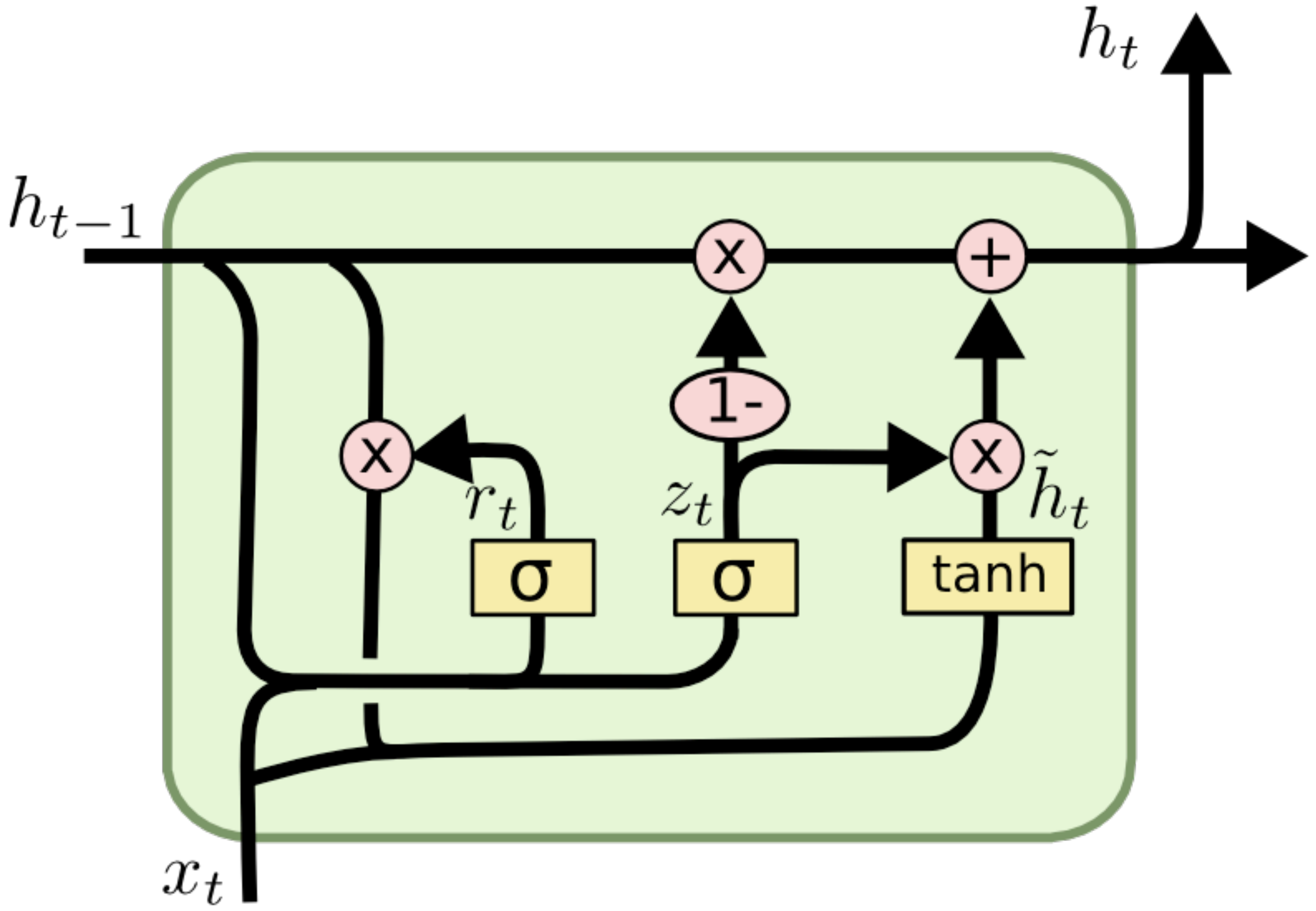
$$C_t = f_t * C_{t-1} + (1 - \boldsymbol{f_t}) * \tilde{C}_t$$

Image Credit: Christopher Olah (http://colah.github.io/posts/2015-08-Understanding-LSTMs/)

* slide from Dhruv Batra

Gated Recurrent Unit (**GRU**)

No explicit memory; memory = hidden output



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

z = memorize new and forget old

Image Credit: Christopher Olah (http://colah.github.io/posts/2015-08-Understanding-LSTMs/)

* slide from Dhruv Batra

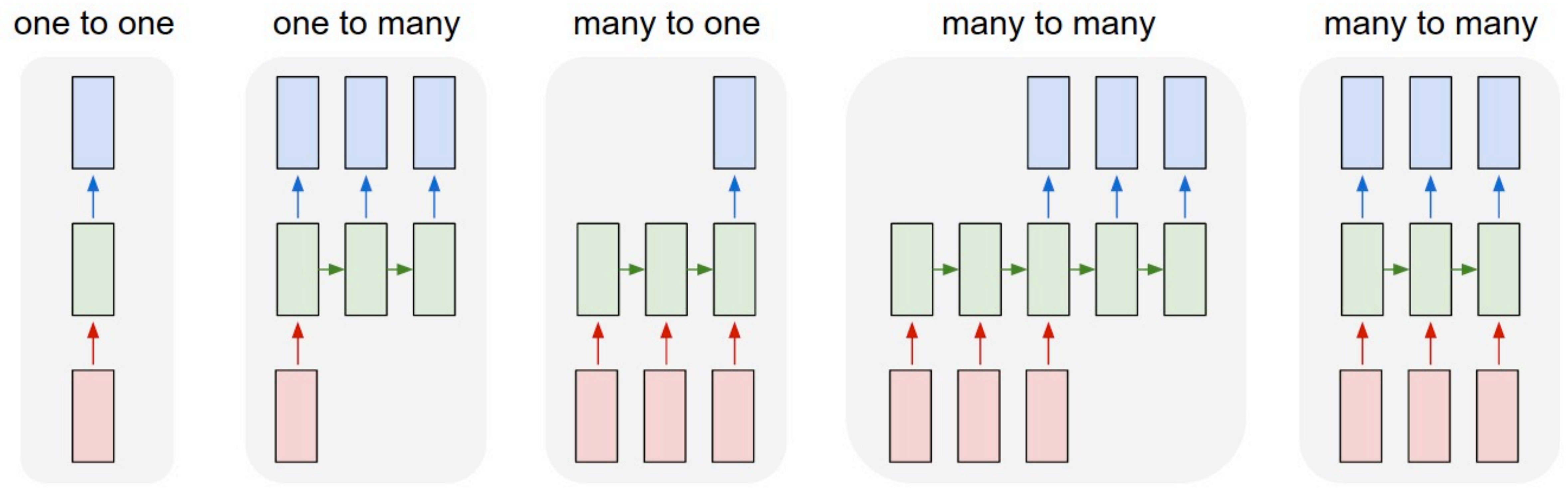
RNNs: Review

Key Enablers:

— Parameter sharing in computational graphs

— “Unrolling” in computational graphs

— Allows modeling **arbitrary length sequences**!



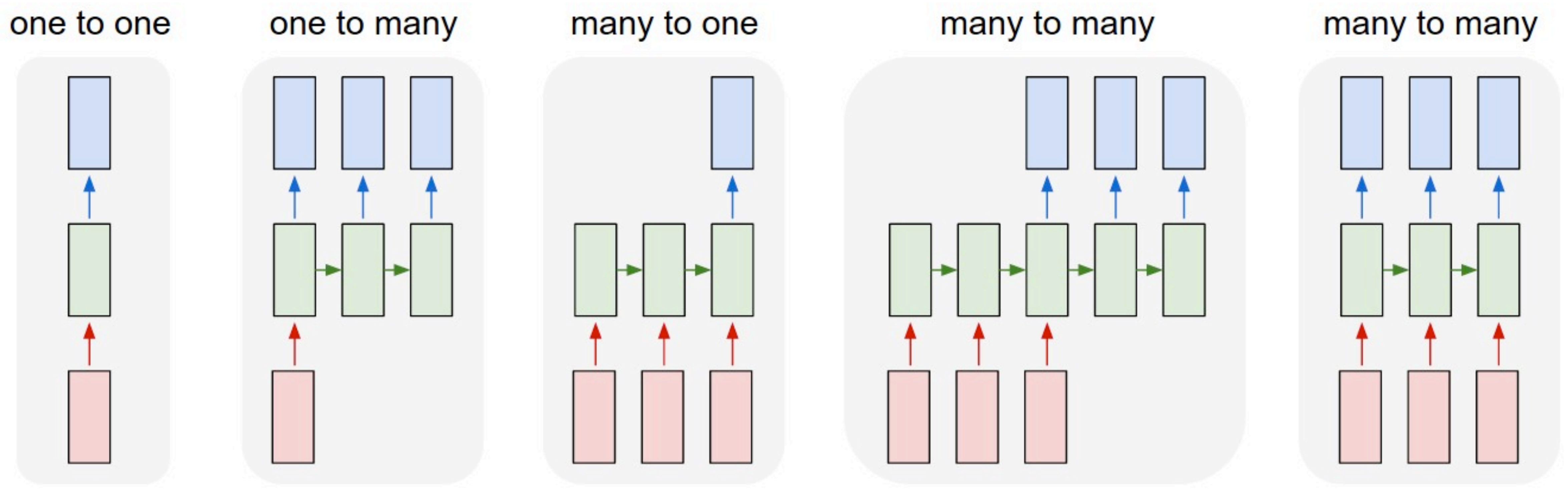
RNNs: Review

Key Enablers:

— Parameter sharing in computational graphs

— “Unrolling” in computational graphs

— Allows modeling **arbitrary length sequences**!



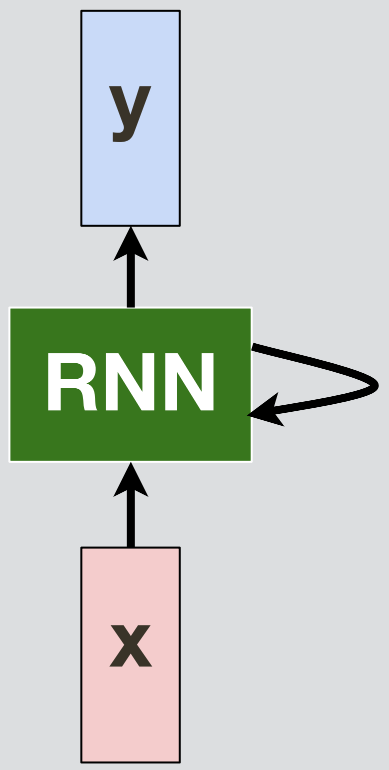
Vanilla **RNN**

$$y_t = W_{hy}h_t + b_y$$

$$h_t = f_W(h_{t-1}, x_t)$$

$$\downarrow$$

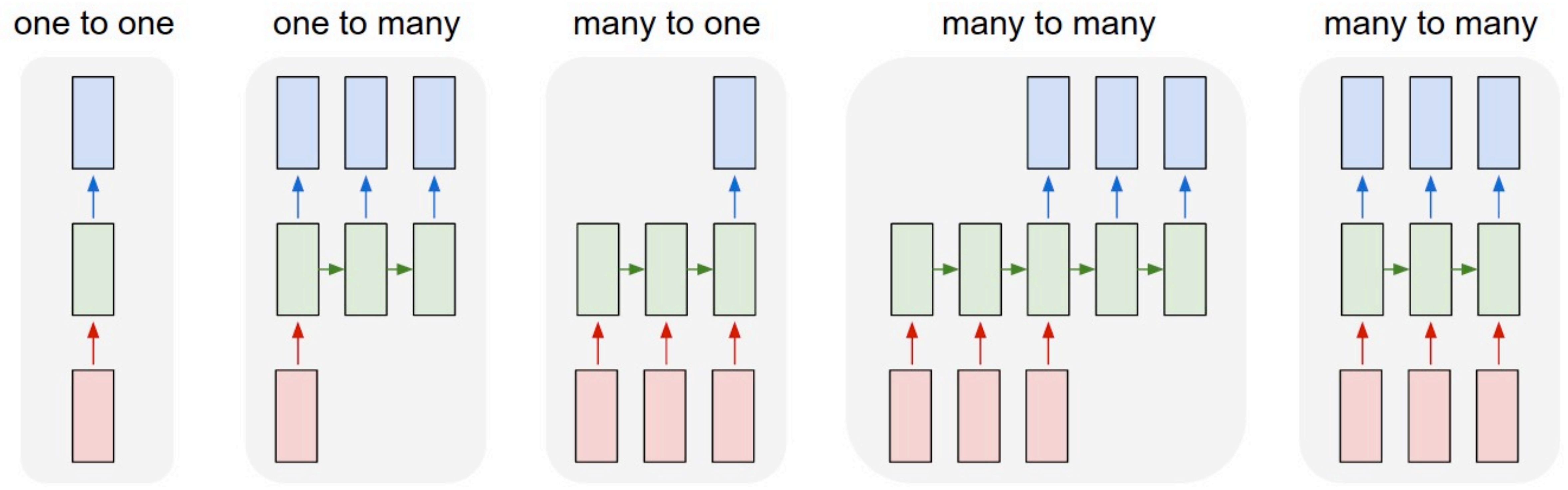
$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t + b_h)$$



RNNs: Review

Key Enablers:

- Parameter sharing in computational graphs
- "Unrolling" in computational graphs
- Allows modeling **arbitrary length sequences**!



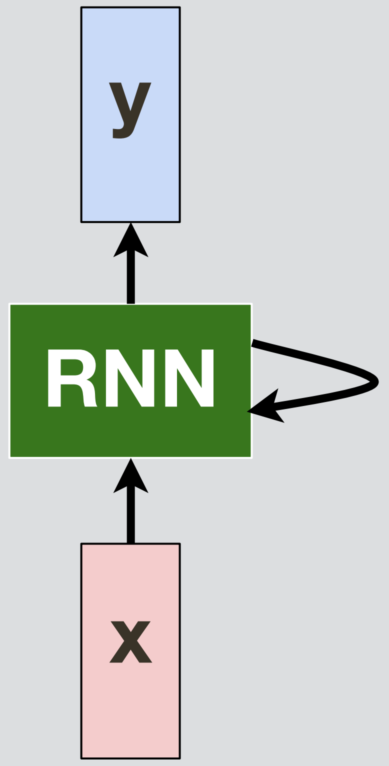
Vanilla RNN

$$y_t = W_{hy} h_t + b_y$$

$$h_t = f_W(h_{t-1}, x_t)$$

$$\downarrow$$

$$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t + b_h)$$



Vanishing or **Exploding** Gradients

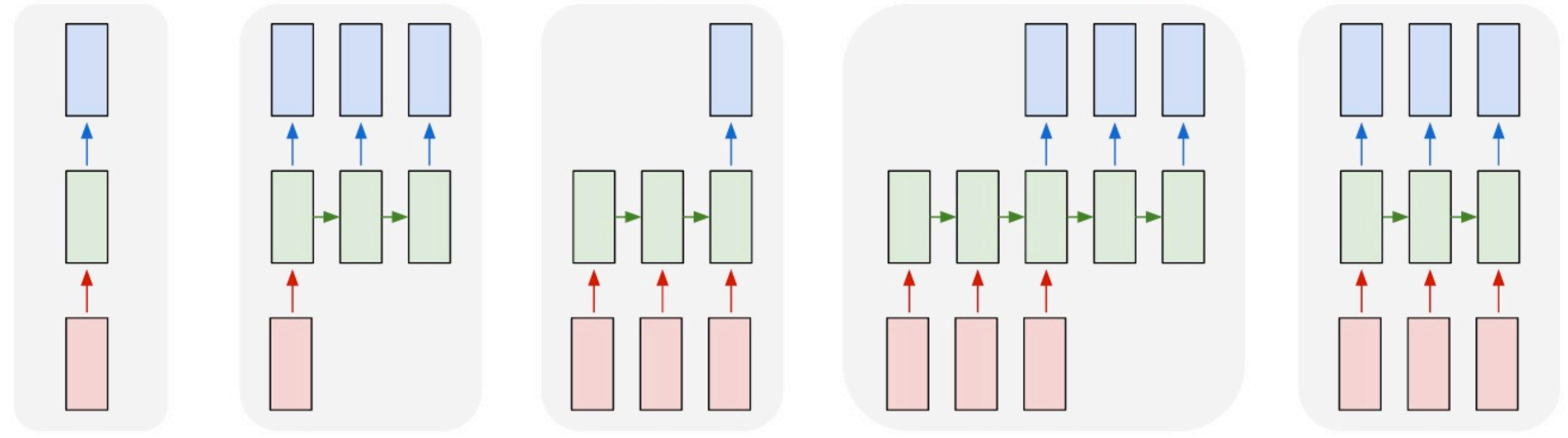
RNNs: Review

Key Enablers:

— Parameter sharing in computational graphs

— "Unrolling" in computational graphs

— Allows modeling **arbitrary length sequences**!



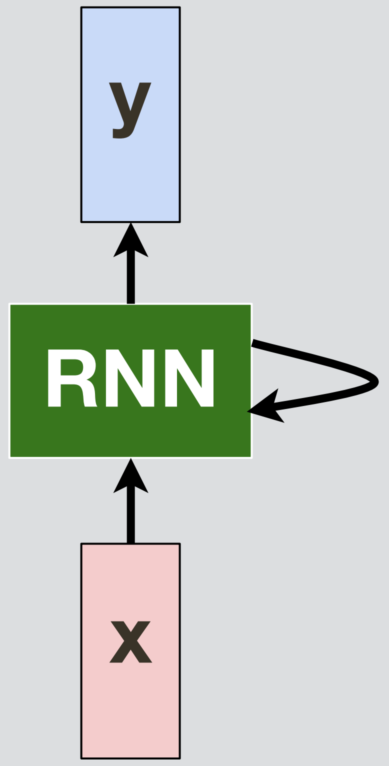
Vanilla **RNN**

$$y_t = W_{hy} h_t + b_y$$

$$h_t = f_W(h_{t-1}, x_t)$$

$$\downarrow$$

$$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t + b_h)$$



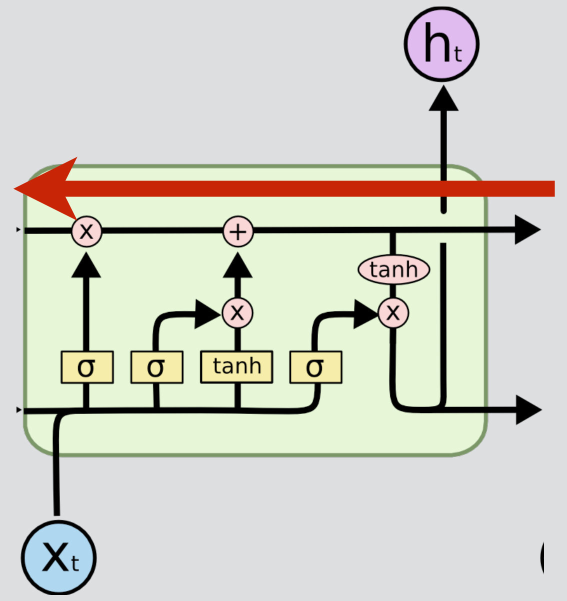
Vanishing or **Exploding** Gradients

Long-Short Term Memory (**LSTM**)

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$



Uninterrupted gradient flow!

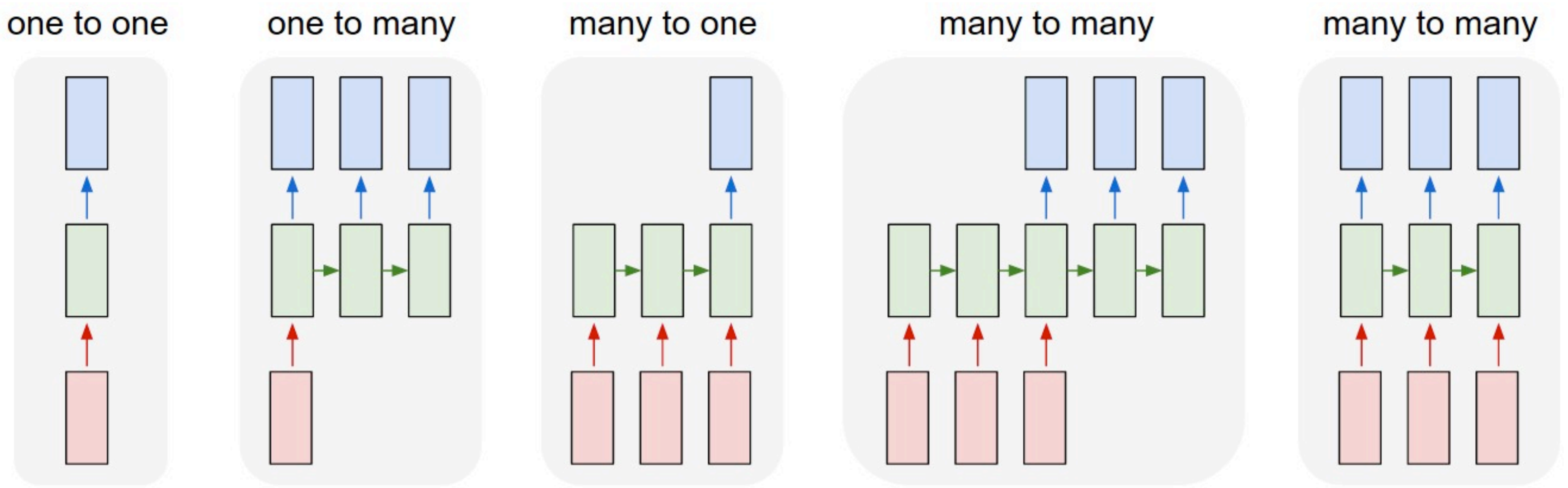
RNNs: Review

Key Enablers:

— Parameter sharing in computational graphs

— “Unrolling” in computational graphs

— Allows modeling **arbitrary length sequences**!



Loss functions: often cross-entropy (for classification); could be max-margin (like in SVM) or Squared Loss (regression)

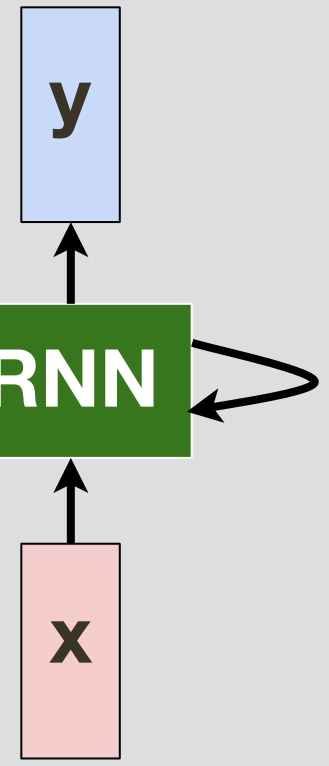
Vanilla **RNN**

$$y_t = W_{hy}h_t + b_y$$

$$h_t = f_W(h_{t-1}, x_t)$$

$$\downarrow$$

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t + b_h)$$



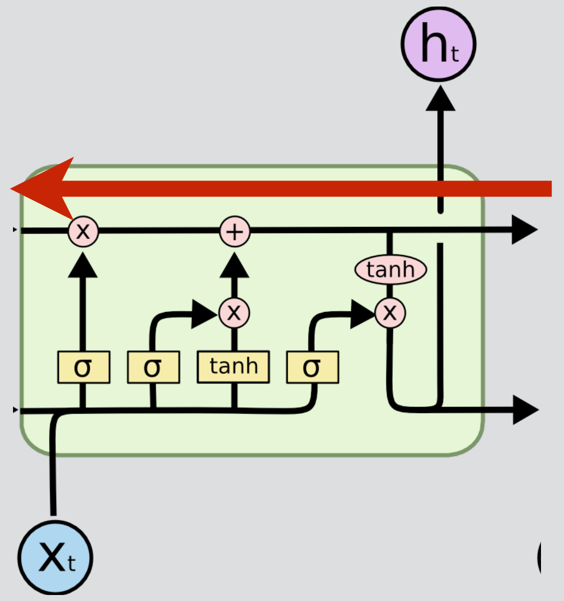
Vanishing
or
Exploding
Gradients

Long-Short Term Memory (**LSTM**)

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

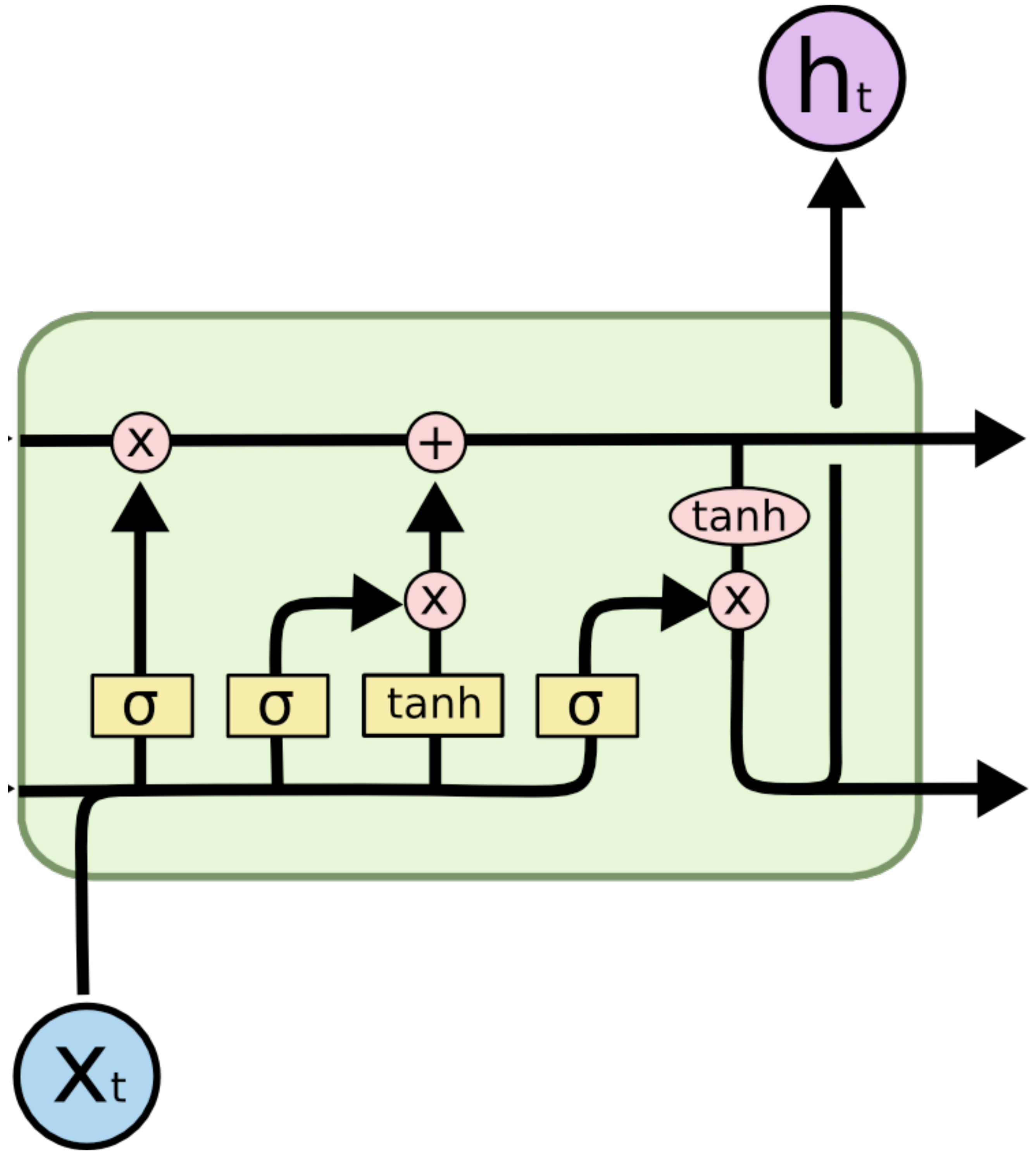
$$h_t = o \odot \tanh(c_t)$$



Uninterrupted gradient flow!

LSTM/RNN **Challenges**

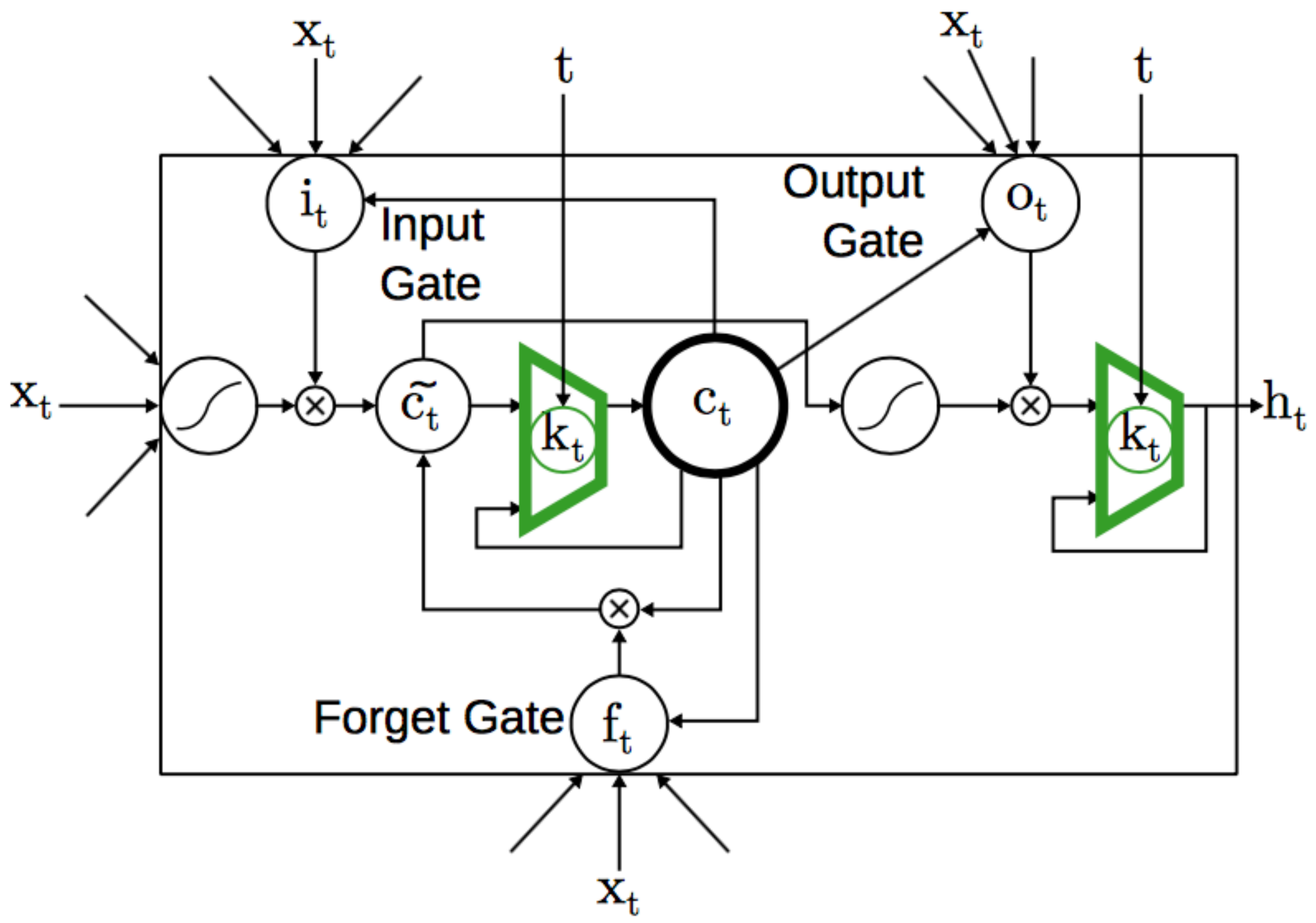
- LSTM can remember some history, but not too long
- LSTM assumes data is regularly sampled

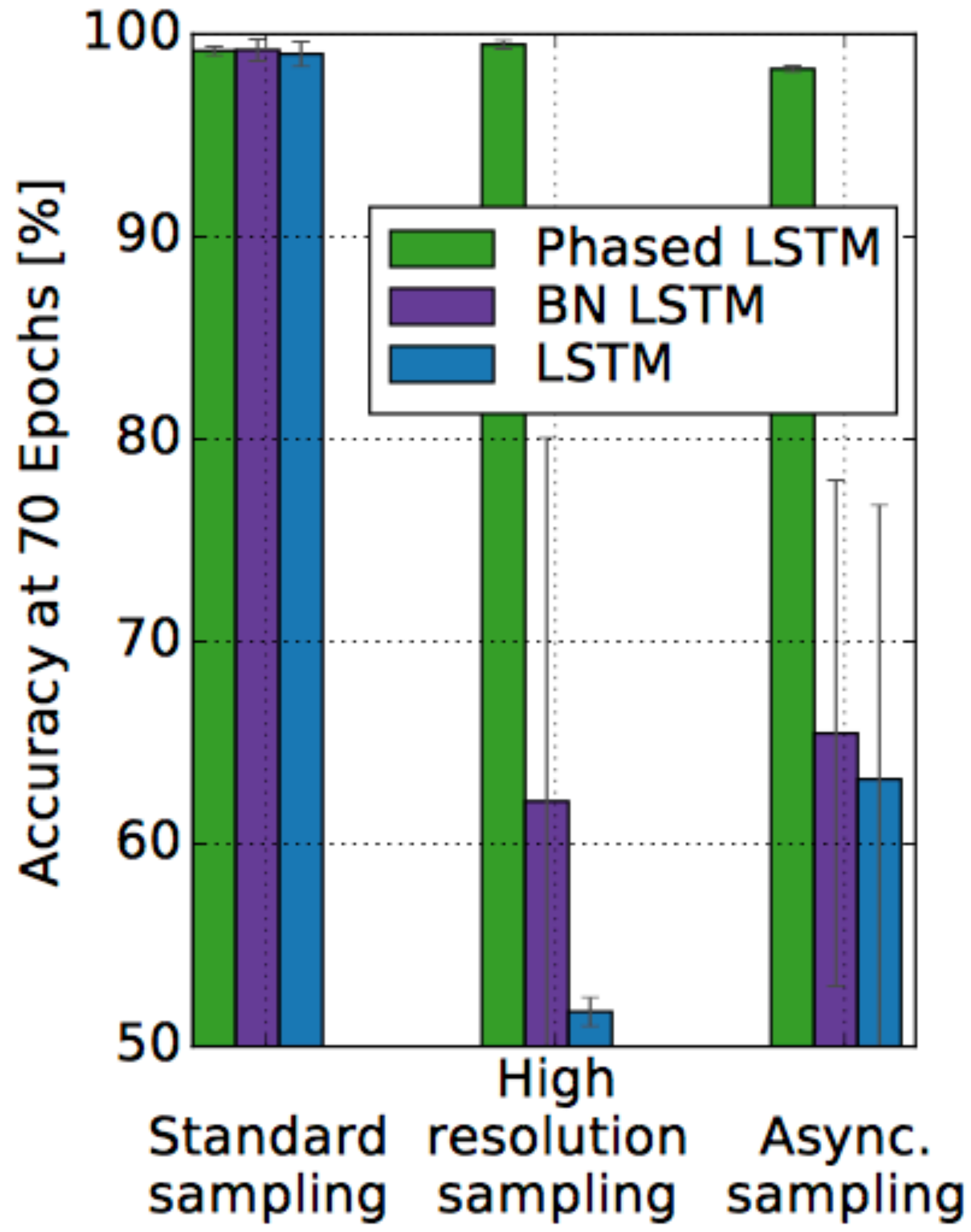


Phased LSTM

[Neil et al., 2016]

Gates are controlled by **phased** (periodic) **oscillations**



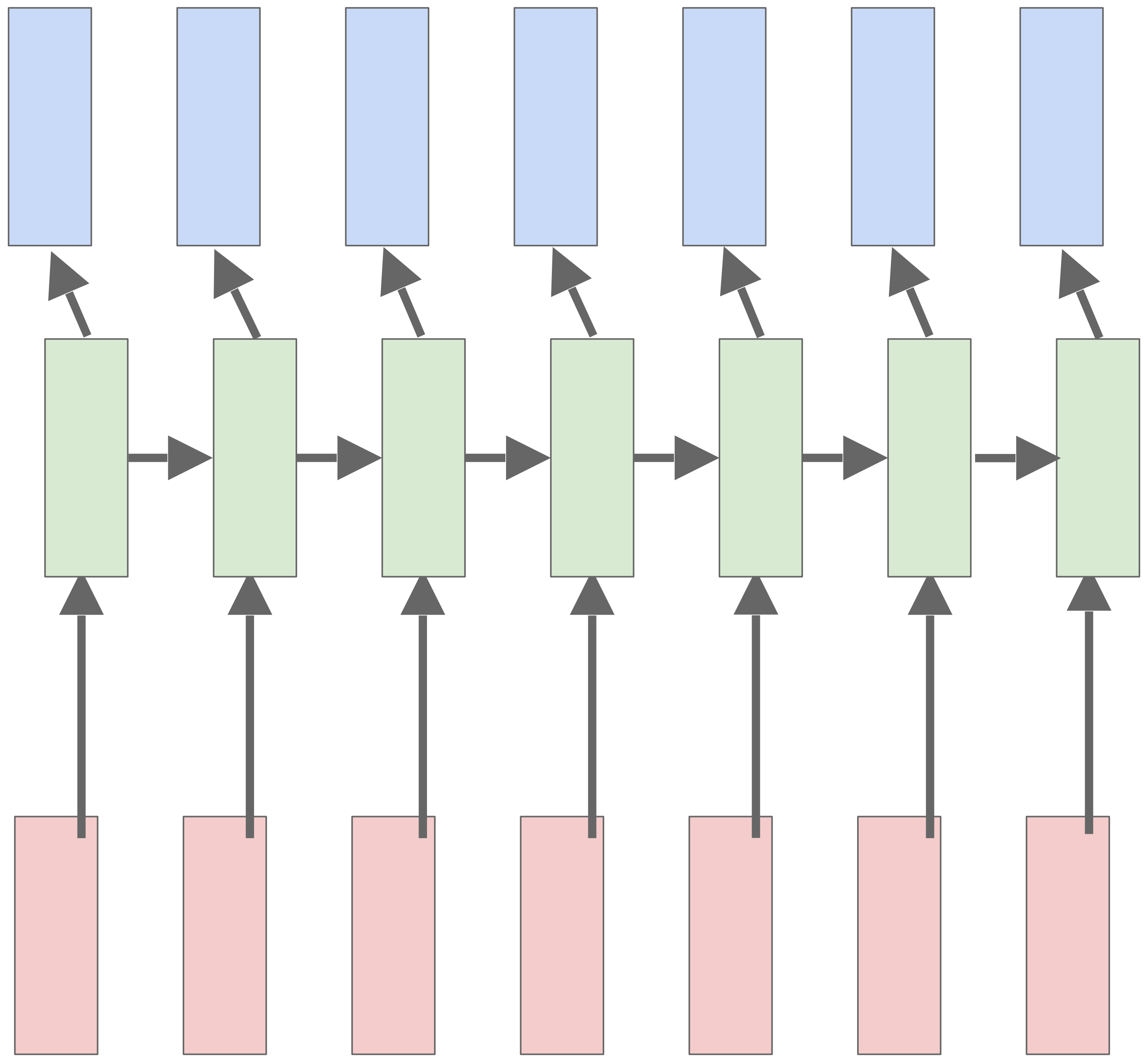


Bi-directional RNNs/LSTMs

$$y_t = W_{hy} h_t + b_y$$

$$h_t = f_W(h_{t-1}, x_t)$$

$$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t + b_h)$$



Bi-directional RNNs/LSTMs

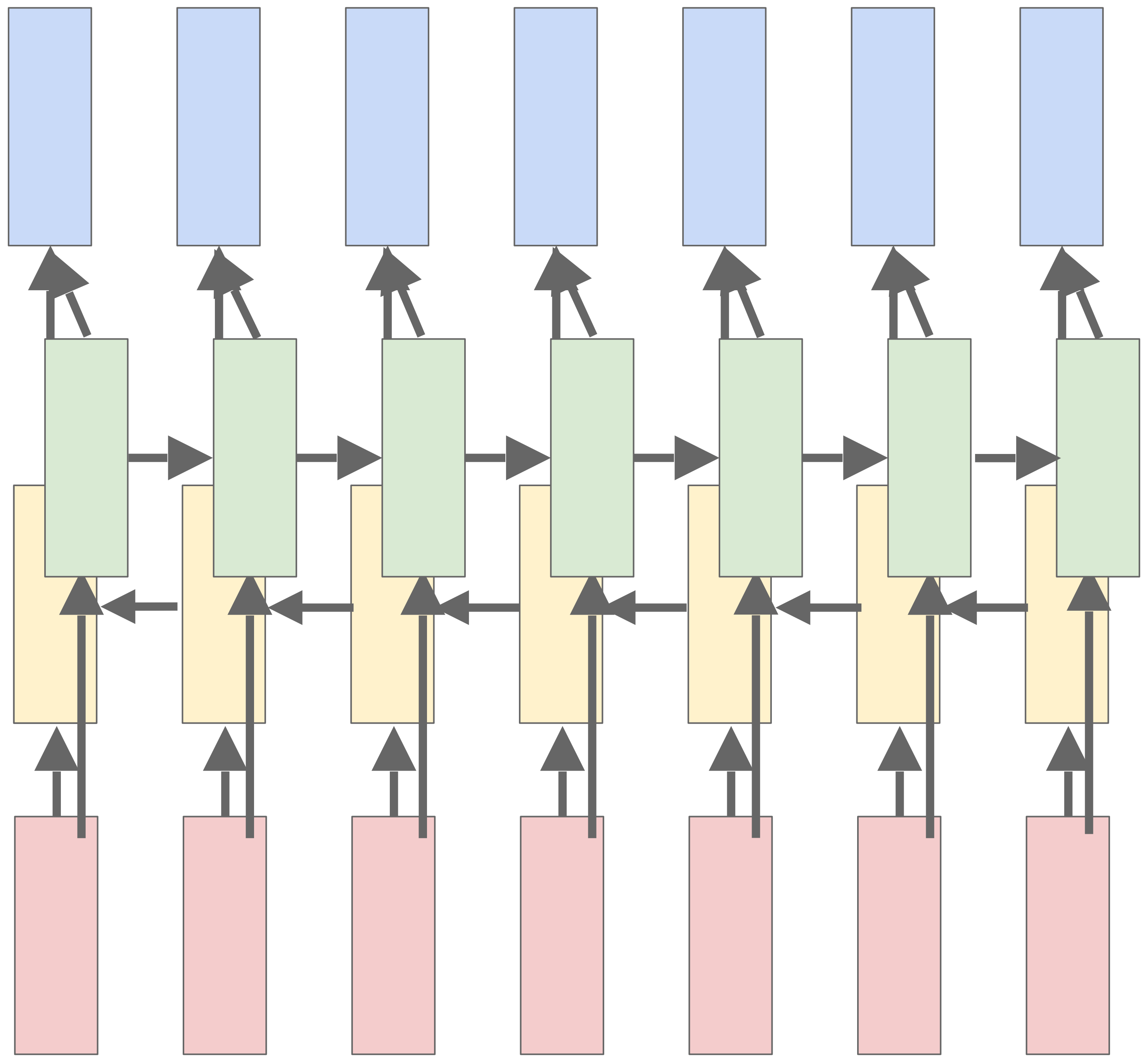
$$y_t = W_{hy}[\overrightarrow{h}_t, \overleftarrow{h}_t]^T + b_y$$

$$\overrightarrow{h}_t = f_{\overrightarrow{W}}(\overrightarrow{h}_{t-1}, x_t)$$

$$\overleftarrow{h}_t = f_{\overleftarrow{W}}(\overleftarrow{h}_{t+1}, x_t)$$

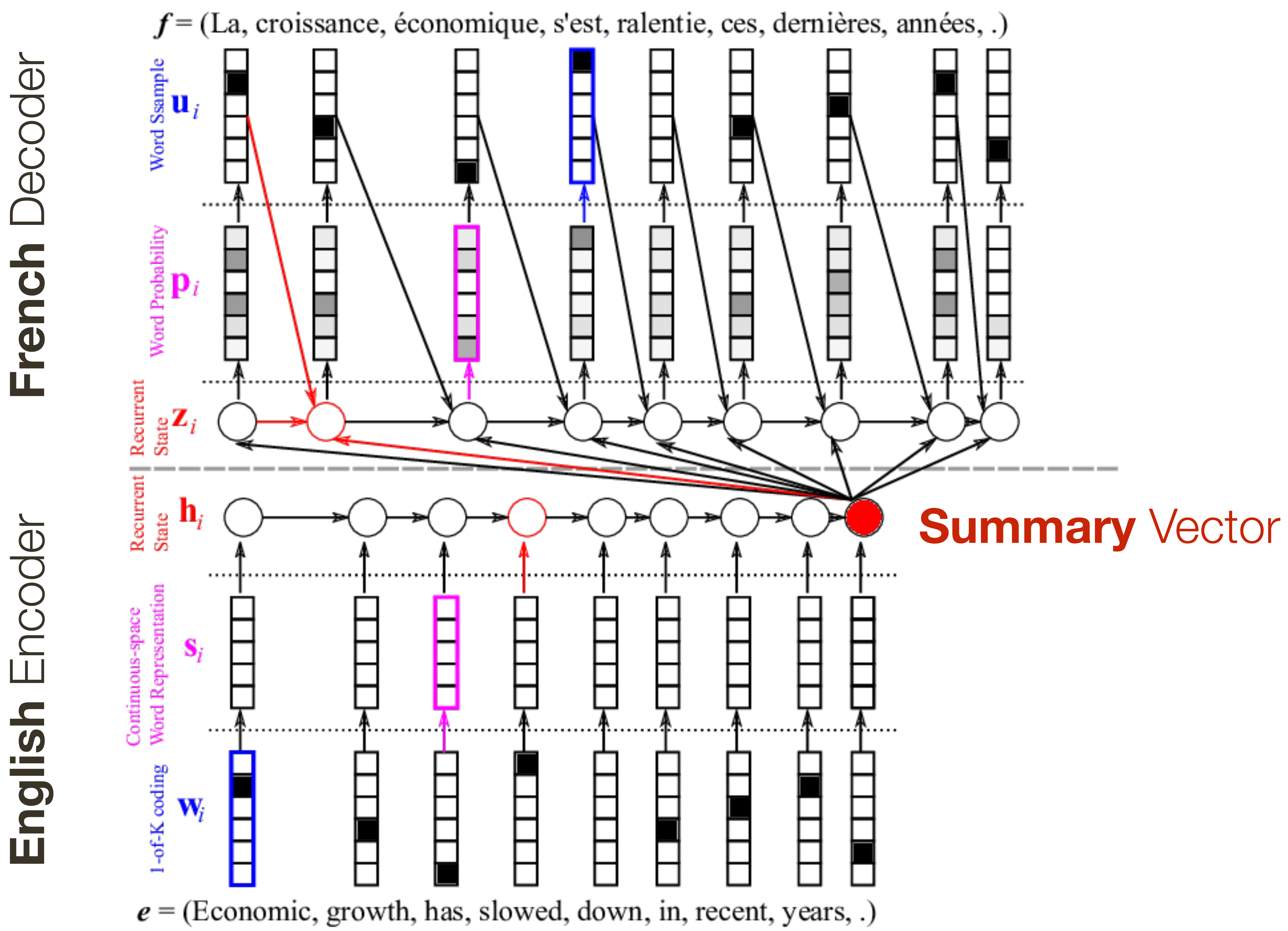
$$\overrightarrow{h}_t = \tanh(\overrightarrow{W}_{hh}\overrightarrow{h}_{t-1} + \overrightarrow{W}_{xh}x_t + \overrightarrow{b}_h)$$

$$\overleftarrow{h}_t = \tanh(\overleftarrow{W}_{hh}\overleftarrow{h}_{t+1} + \overleftarrow{W}_{xh}x_t + \overleftarrow{b}_h)$$



Attention Mechanisms and RNNs

Consider a **translation task**: This is one of the first neural translation models

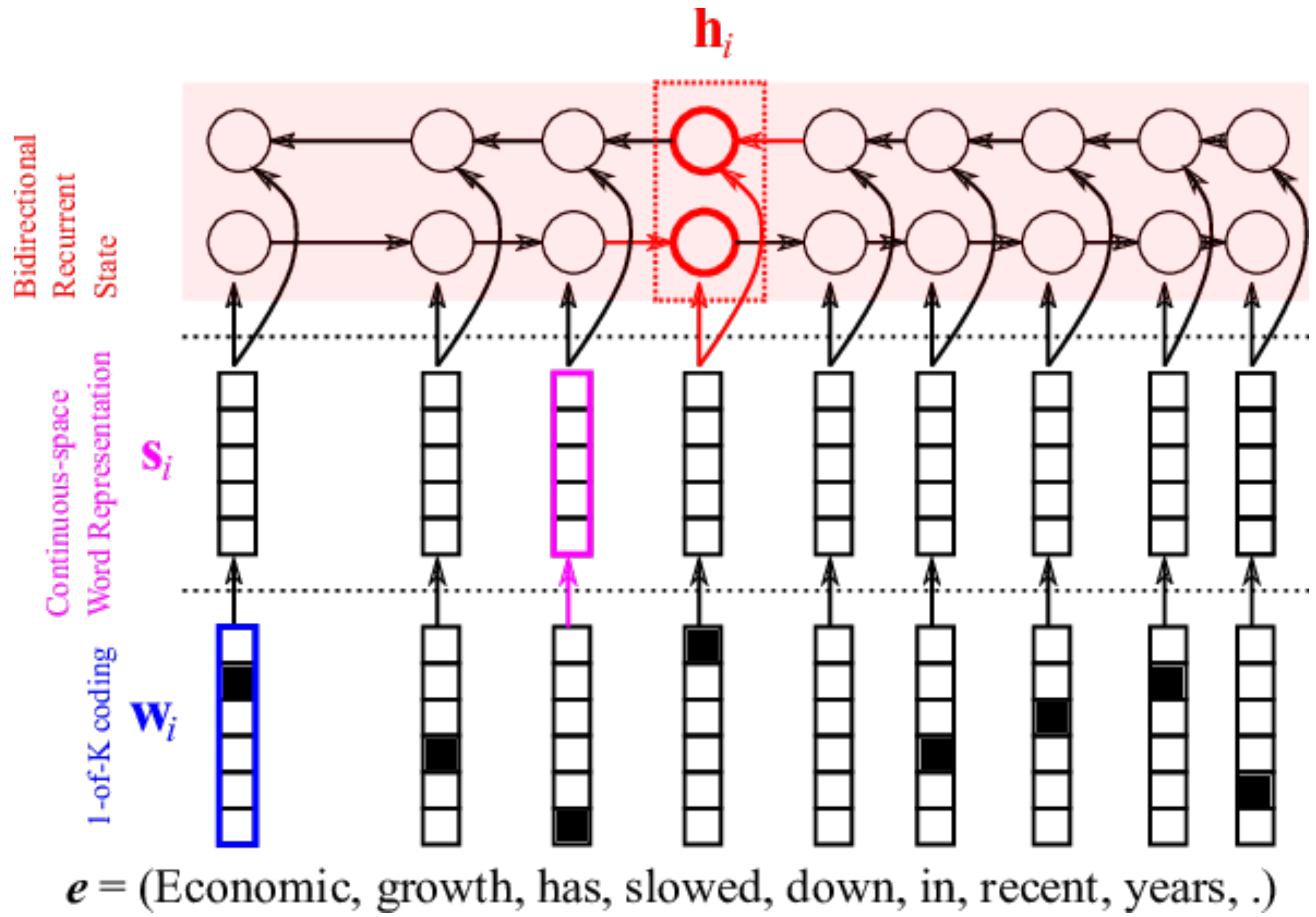


https://devblogs.nvidia.com/introduction-neural-machine-translation-gpus-part-3/

Attention Mechanisms and RNNs

Consider a **translation task** with a bi-directional encoder of the source language

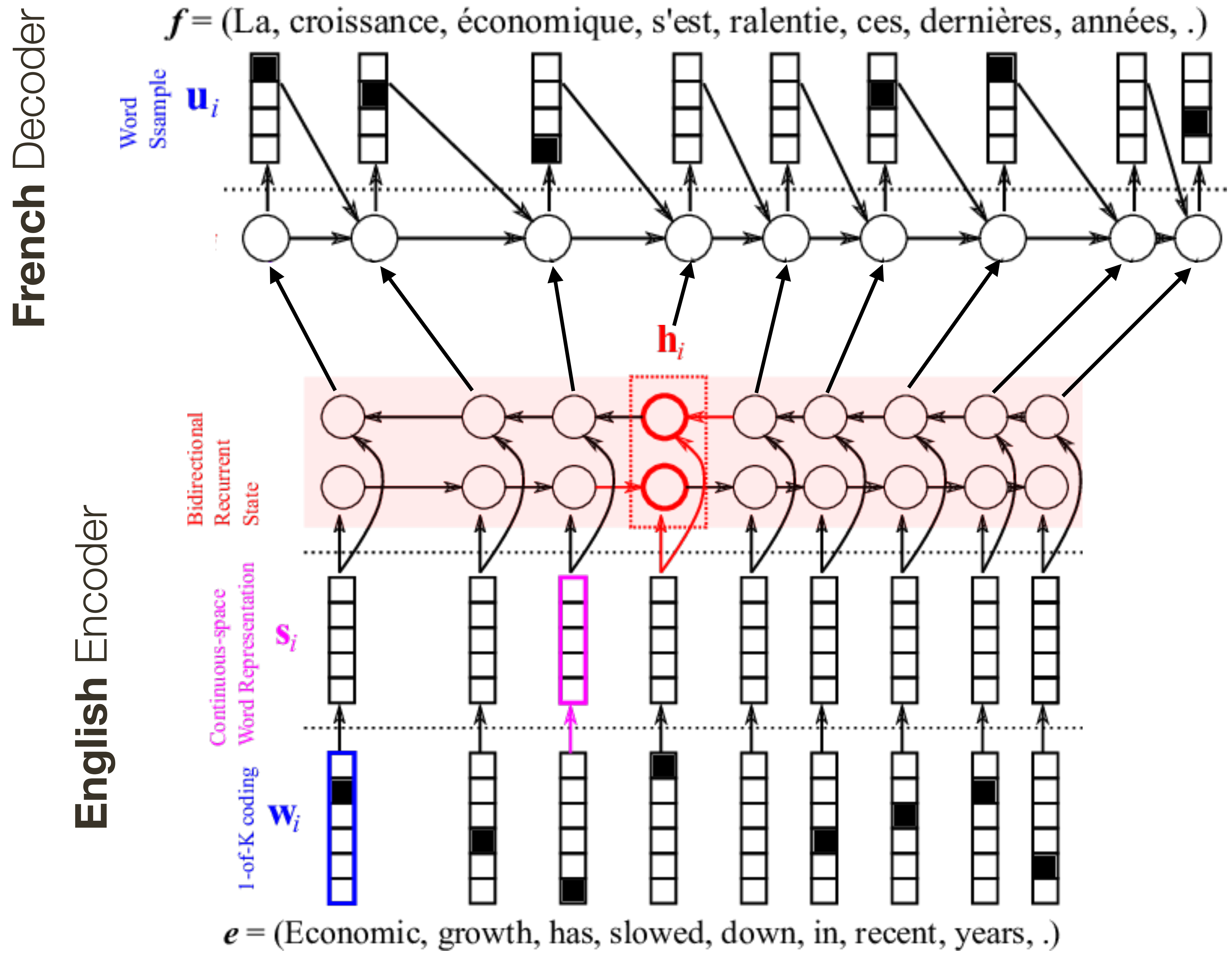
English Encoder



https://devblogs.nvidia.com/introduction-neural-machine-translation-gpus-part-3/

Attention Mechanisms and RNNs

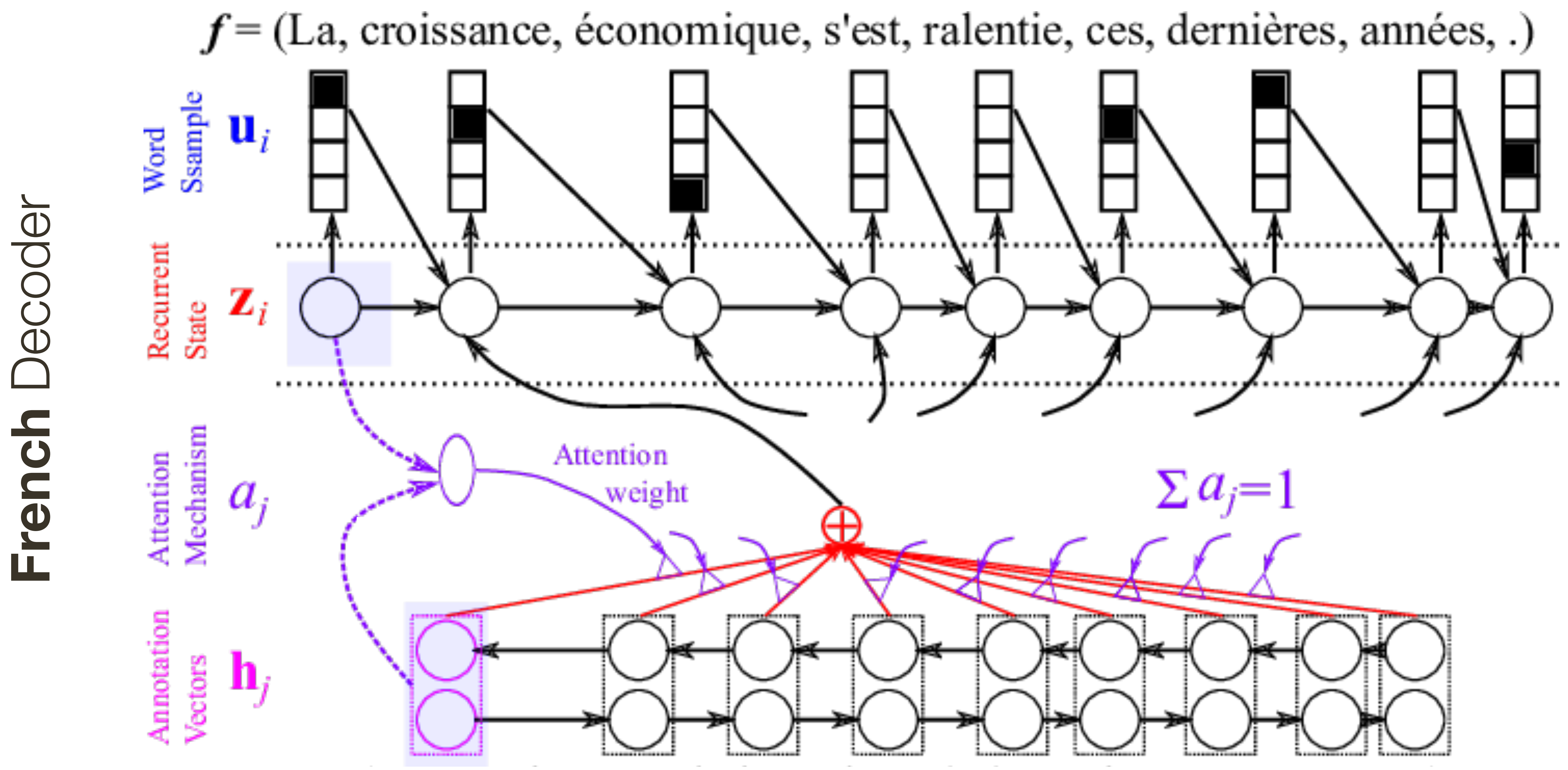
Consider a **translation task** with a bi-directional encoder of the source language



https://devblogs.nvidia.com/introduction-neural-machine-translation-gpus-part-3/

Attention Mechanisms and RNNs

Consider a **translation task** with a bi-directional encoder of the source language



[Cho et al., 2015]

Build a **small neural network** (one layer) with softmax output that takes

(1) everything decoded so far and (encoded by previous decoder state Zi)

(2) encoding of the current word (encoded by the hidden state of encoder hj)

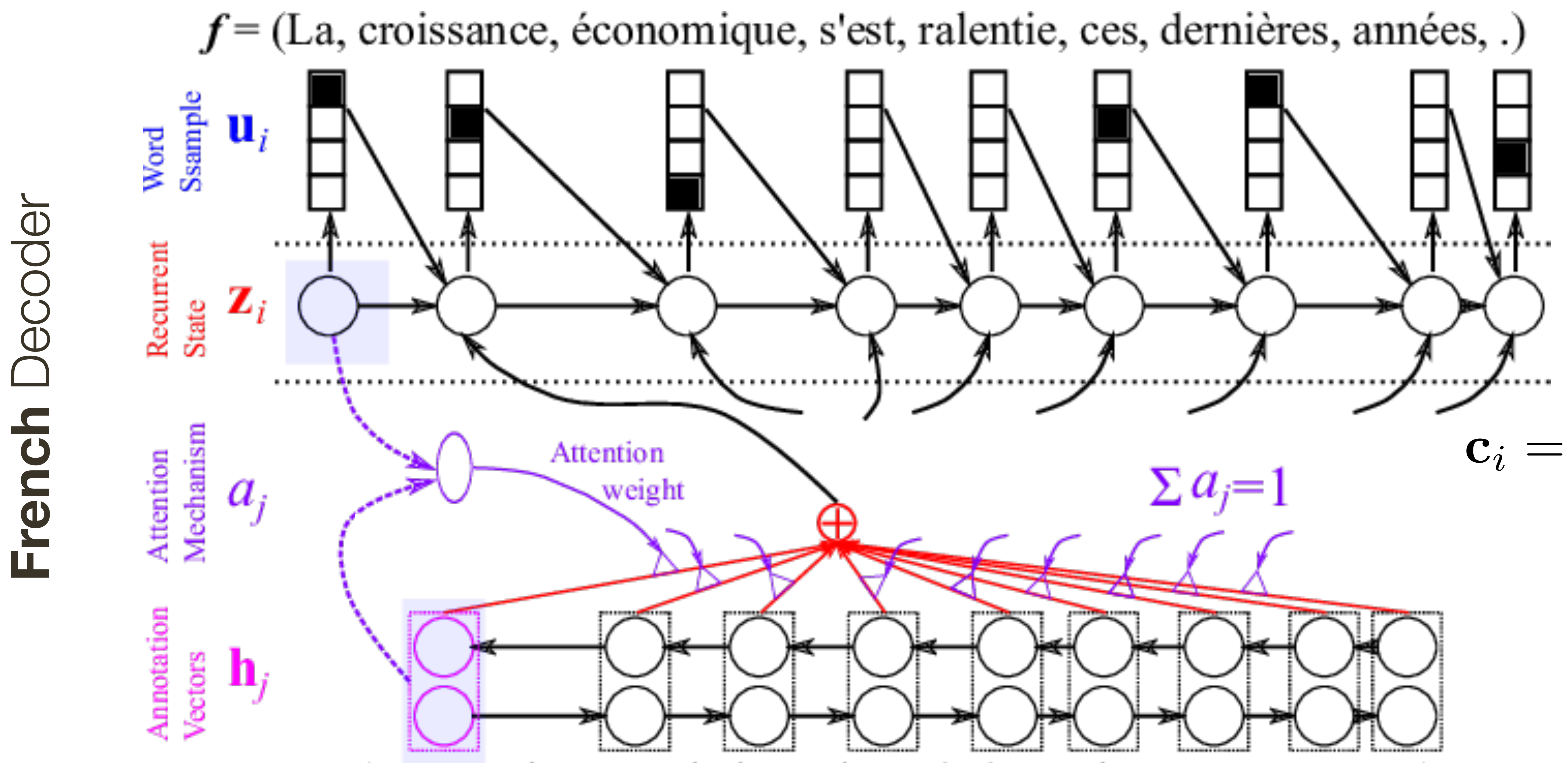
and predicts **relevance of every source word** towards next translation

https://devblogs.nvidia.com/introduction-neural-machine-translation-gpus-part-3/

Attention Mechanisms and RNNs

Consider a **translation task** with a bi-directional encoder of the source language

[Cho et al., 2015]



$$\mathbf{c}_i = \sum_{j=1}^{T} \alpha_j \mathbf{h}_j$$

Build a **small neural network** (one layer) with softmax output that takes

(1) everything decoded so far and (encoded by previous decoder state Zi)

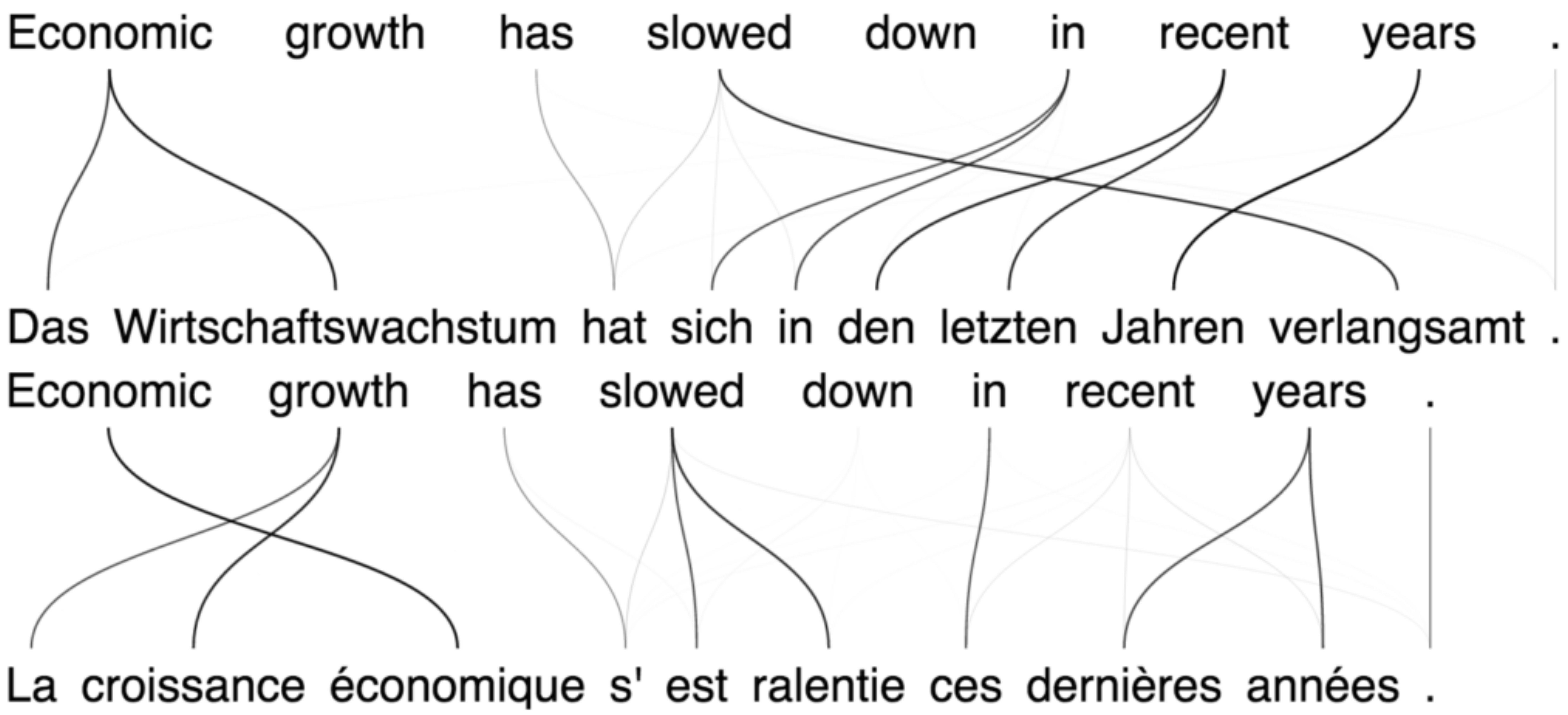
(2) encoding of the current word (encoded by the hidden state of encoder hj)

and predicts **relevance of every source word** towards next translation

https://devblogs.nvidia.com/introduction-neural-machine-translation-gpus-part-3/

Attention Mechanisms and RNNs

[Cho et al., 2015]



https://devblogs.nvidia.com/introduction-neural-machine-translation-gpus-part-3/

Regularization in RNNs

Standard dropout in recurrent layers does not work because it causes **loss of long term memory**!

* slide from Marco Pedersoli and Thomas Lucas

Regularization in RNNs

Standard dropout in recurrent layers does not work because it causes **loss of long term memory**!

—Dropout in input-to-hidden or hidden-to-output layers [Zaremba et al., 2014]

— Apply dropout at sequence level (same zeroed units for the entire sequence) [Gal, 2016]

— Dropout only at the cell update (for LSTM and GRU units) [Semeniuta et al., 2016]

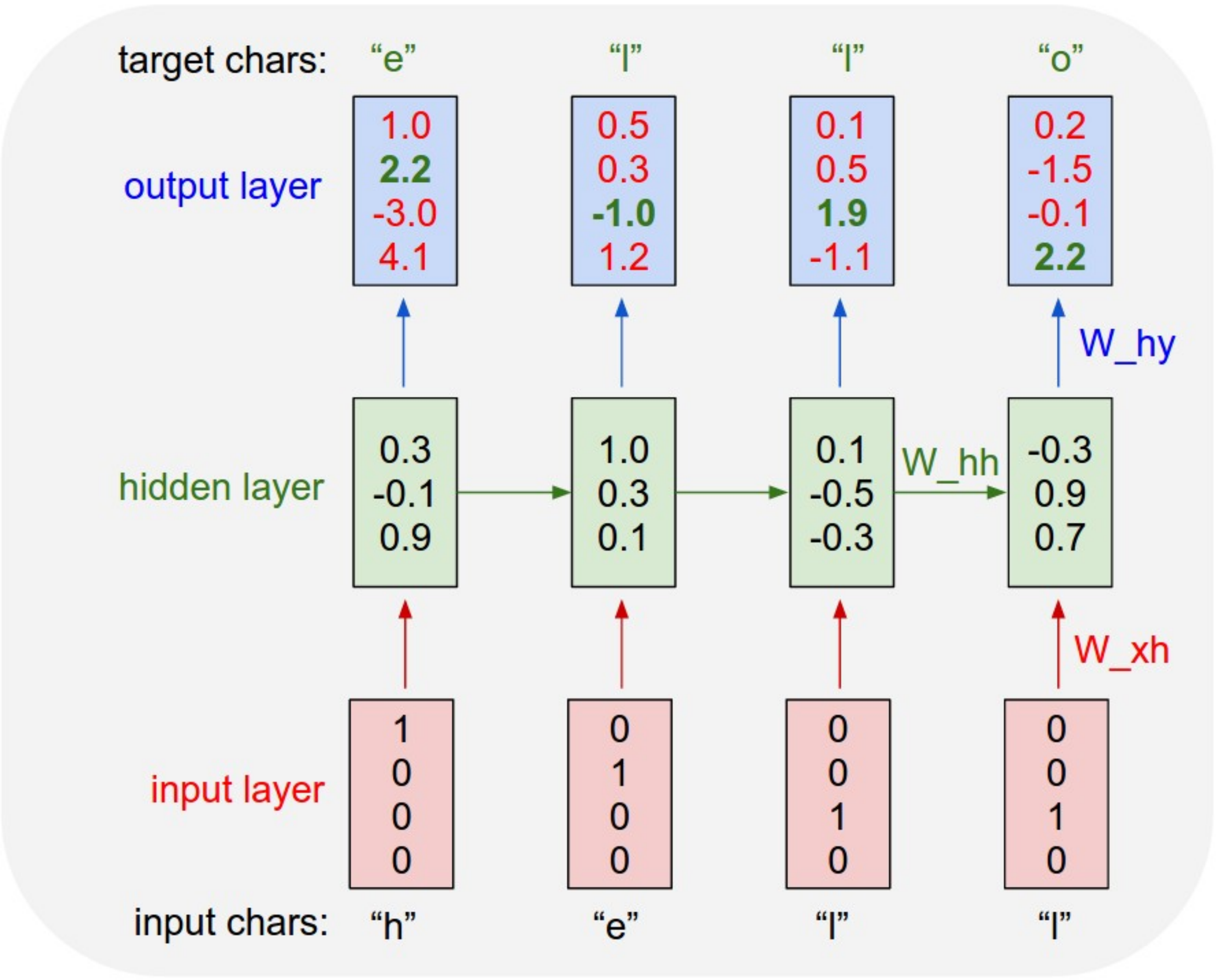
— Enforcing norm of the hidden state to be similar along time [Krueger & Memisevic, 2016]

— Zoneout some hidden units (copy their state to the next tilmestep) [Krueger et al., 2016]

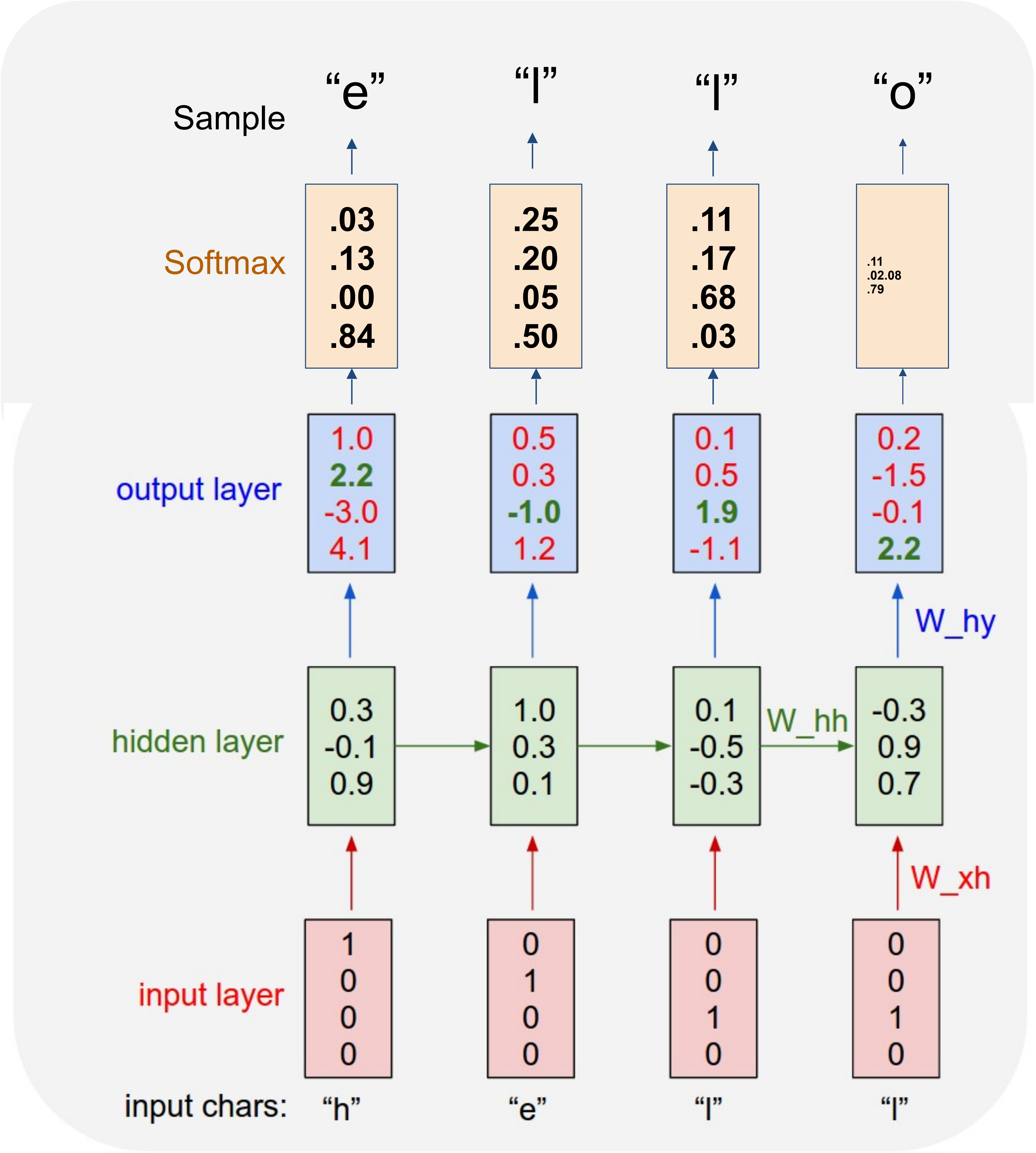
* slide from Marco Pedersoli and Thomas Lucas

Teacher Forcing

Training Objective: Predict the next word (cross entropy loss)



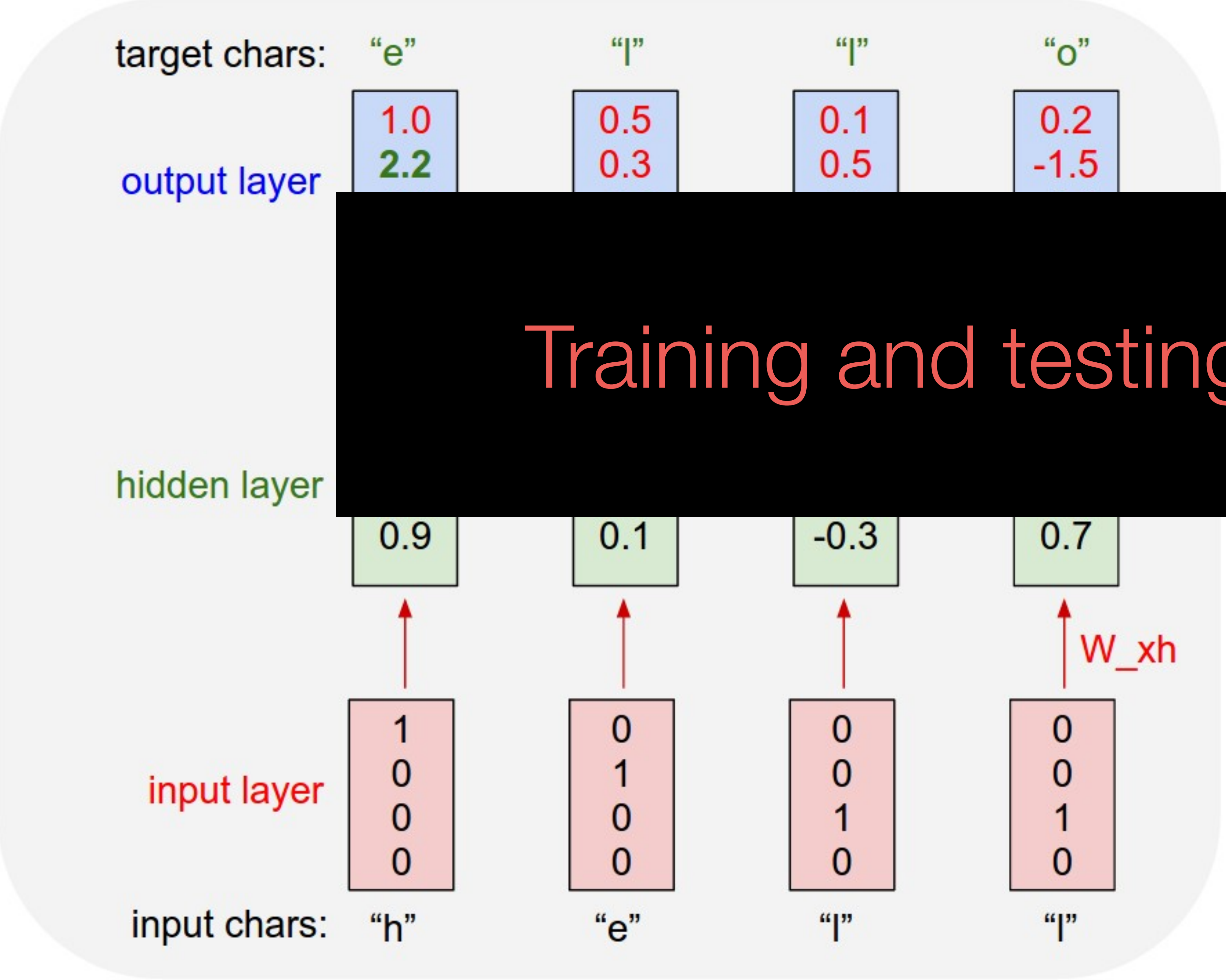
Testing: Sample the full sequence

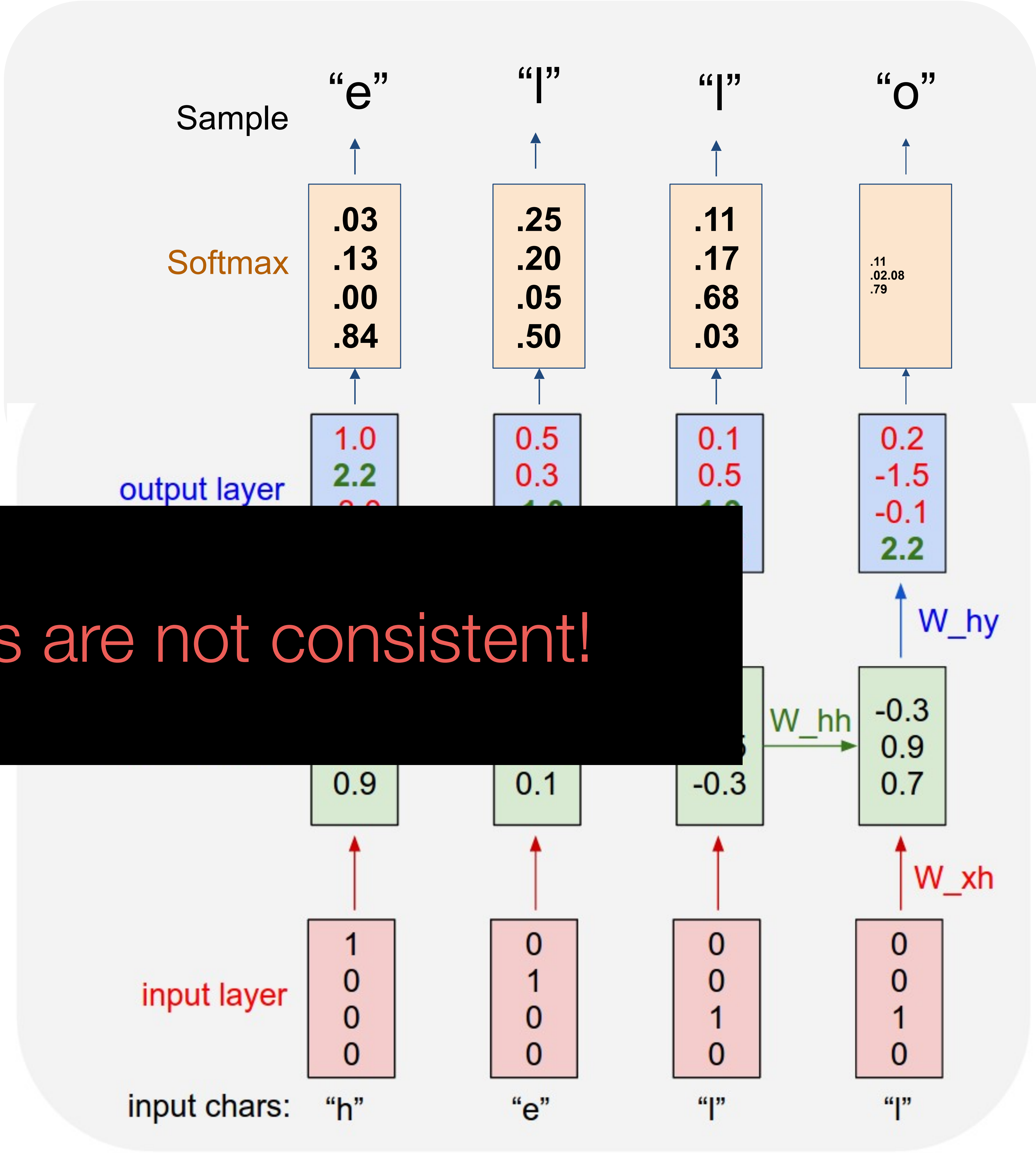


Teacher Forcing

Testing: Sample the full sequence

Training Objective: Predict the next word (cross entropy loss)

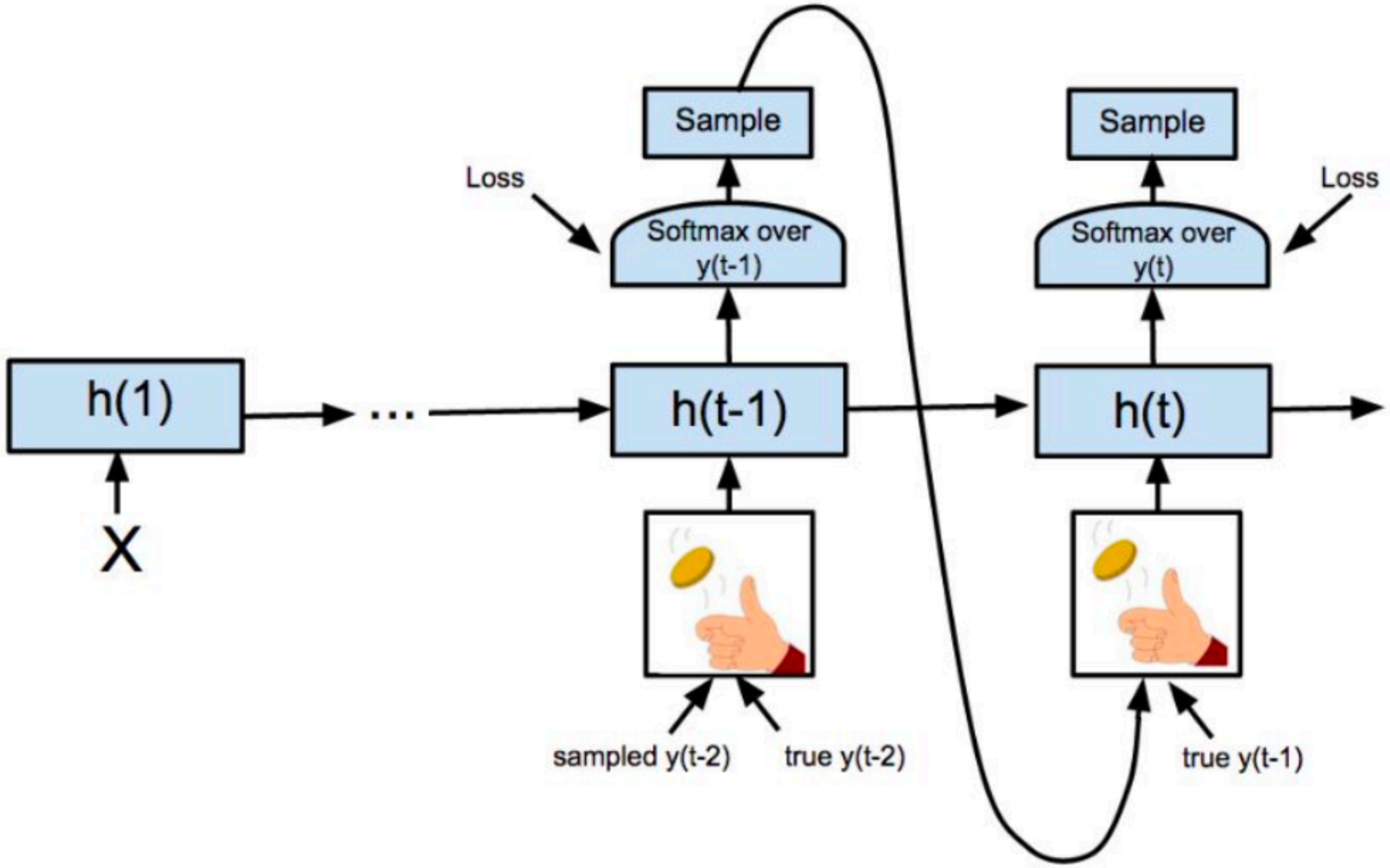


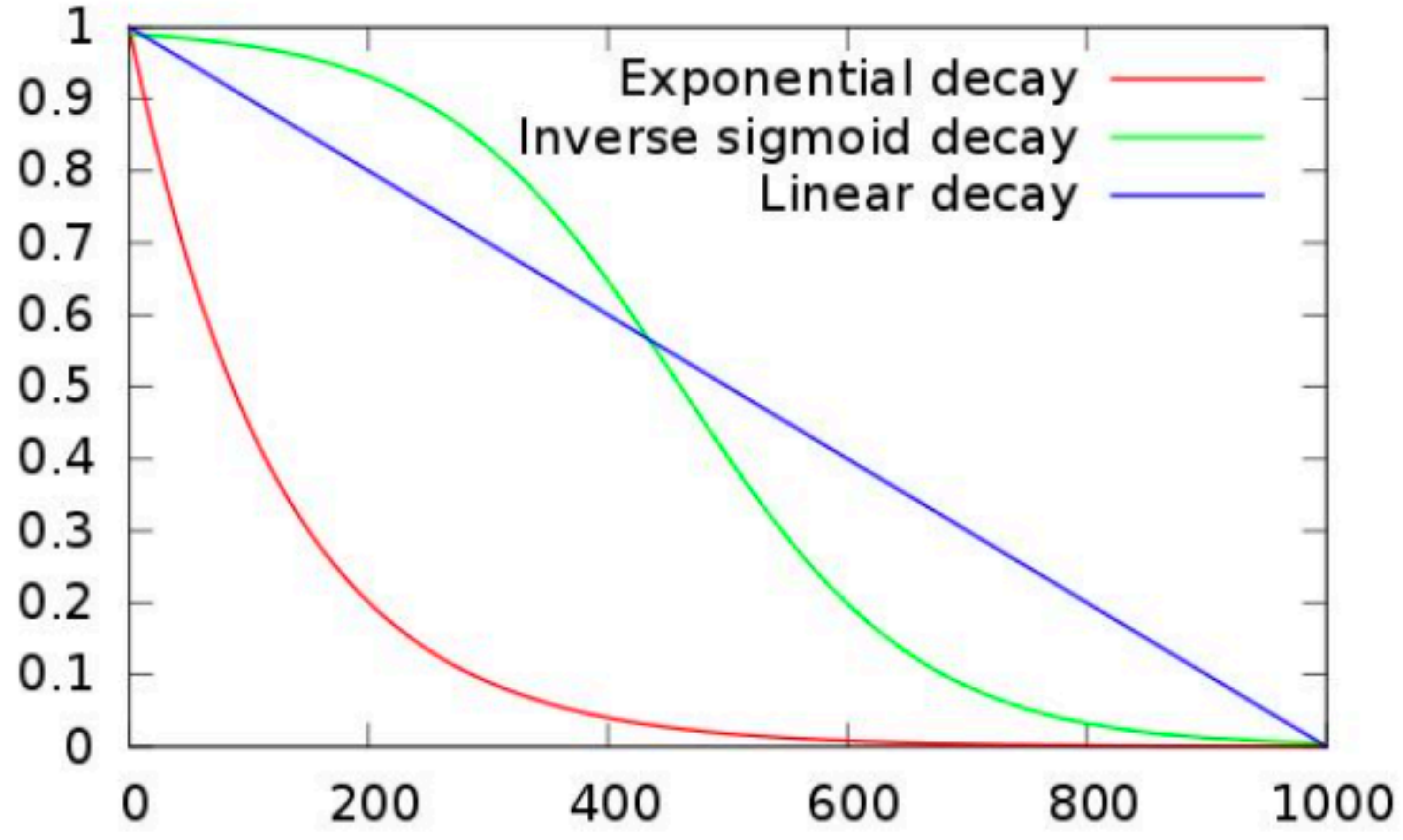


Training and testing objectives are not consistent!

Teacher Forcing

Slowly move from *Teacher Forcing* to *Sampling*





Probability of sampling from the ground truth

[Bengio et al., 2015]

* slide from Marco Pedersoli and Thomas Lucas

Teacher Forcing

Microsoft COCO developement set

Approach vs Metric	BLEU-4	METEOR	CIDER
Baseline	28.8	24.2	89.5
Baseline with Dropout	28.1	23.9	87.0
Always Sampling	11.2	15.7	49.7
Scheduled Sampling	**30.6**	**24.3**	**92.1**
Uniform Scheduled Sampling	29.2	24.2	90.9
Baseline ensemble of 10	30.7	25.1	95.7
Scheduled Sampling ensemble of 5	**32.3**	**25.4**	**98.7**

Baseline: Google NIC captioning model

Baseline **with Dropout**: Regularized RNN version

Always sampling: Use sampling from the beginning of training

Scheduled sampling: Sampling with inverse Sigmoid decay

Uniformed scheduled sampling: Scheduled sampling but uniformly

* slide from Marco Pedersoli and Thomas Lucas

Sequence Level Training

During training objective is different than at test time

- **Training:** generate next word given the previous
- **Test:** generate the entire sequence given an initial state

Optimize directly evaluation metric (e.g. BLUE score for sentence generation)

Set the problem as a Reinforcement Learning:

- RNN is an Agent
- Policy defined by the learned parameters
- Action is the selection of the next word based on the policy - Reward is the evaluation metric

[Ranzato et al., 2016]

* slide from Marco Pedersoli and Thomas Lucas